AFRL-RI-RS-TR-2013-254

INVESTIGATING COGNITIVE RHYTHMS AS A NEW MODALITY FOR CONTINUOUS AUTHENTICATION

NEW YORK INSTITUTE OF TECHNOLOGY

DECEMBER 2013

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2013-254 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /	**/ S /**
TODD CUSHMAN Work Unit Manager	WARREN H. DEBANY, JR. Technical Advisor, Information Exploitation and Operations Division Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* DECEMBER 2013	**2. REPORT TYPE** FINAL TECHNICAL REPORT	**3. DATES COVERED** *(From - To)* JUN 2012 – MAY 2013

4. TITLE AND SUBTITLE

INVESTIGATING COGNITIVE RHYTHMS AS A NEW MODALITY FOR CONTINUOUS AUTHENTICATION

5a. CONTRACT NUMBER
FA8750-12-2-0201

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
61101E

6. AUTHOR(S)

Kiran S. Balagani

5d. PROJECT NUMBER
ATAU

5e. TASK NUMBER
NY

5f. WORK UNIT NUMBER
IT

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
New York Institute of Technology
School of Engineering and Computer Sciences
Harry Schure Hall, Room 226c
Northern Boulevard
Old Wesbury NY 11568-8000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research
Projects Agency
3701 North Fairfax Drive
Arlington, VA 22203-1714

Air Force Research Laboratory/RIGA
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2013-254

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Research performed and results achieved under the DARPA Active Authentication award “Investigating Cognitive Rhythms as a New Modality for Continuous Authentication.” The research under this project focused on five main tasks: (1) defining and extracting cognitive rhythms (atomic, pausality, linguistic, and revision features) from users’ text production data; (2) determining the availability and discriminability of cognitive rhythm features for continuous authentication; (3) determining continuous authentication accuracies of cognitive rhythm features over a large population of users; (4) investigating the impact of non-zero effort forgery attacks against continuous authentication; and (5) prediction of a users’ cognitive load levels and demographic information.

15. SUBJECT TERMS
Active Authentication, Cognitive Rhythms, Biometrics

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** **TODD CUSHMAN**
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UU	60	**19b. TELEPONE NUMBER (*Include area code)*** **N/A**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

We sincerely thank Mr. Richard Guidorrizi, Program Manager, DARPA, for providing the opportunity and support to conduct research under the auspices of DARPA Active Authentication program. We sincerely thank Mr. Todd Cushman, Air Force Research Laboratory, Ms. Anna Weeks, Air Force Research Laboratory, Mr. Larry Blankenship, DARPA, and Ms. Susan Kloss, DARPA, for their help and support throughout the course of this project. Last but not the least, we are very grateful to all the volunteers who provided typing data at Louisiana Tech University. Without them, our research under this project would not have been possible.

1 SUMMARY

In this document, we report the research performed and results achieved under the DARPA Active Authentication award "Investigating Cognitive Rhythms as a New Modality for Continuous Authentication." Our research under this project focused on five main tasks: (1) defining and extracting cognitive rhythms (atomic, pausality, linguistic, and revision features) from users' text production data; (2) determining the availability and discriminability of cognitive rhythm features for continuous authentication; (3) determining continuous authentication accuracies of cognitive rhythm features over a large population of users; (4) investigating the impact of non-zero effort forgery attacks against continuous authentication; and (5) prediction of a users' cognitive load levels and demographic information. Our key achievements, findings, and results are listed below.

(1) We designed and developed more than 45 types of atomic, pausality, linguistic, and revision feature modules that have enabled us to extract over 9000 individual features;

(2) We performed availability and discriminability analysis of features. The features with highest availability were atomic level features (especially, key hold and digraph latencies). Authentication results confirm that these features, especially combination of hold and digraph latencies, provide high discriminability;

(3) Among the tested features and matching algorithms, we achieved the highest verification (1-to-1 match) and identification (1-to-N match) accuracies with score-level fusion, which included fusing scaled Euclidean, scaled Manhattan, and Relative "R" distance verifiers operating on key hold and digraph latencies;

(4) Our experiments with non-zero effort "snoop-forge-replay" attacks show that continuous verification systems relying on atomic features are susceptible to algorithmic forgeries created from snooped (or stolen) keystrokes.

(5) Our experiments with "frog-boiling" attacks on keystroke based continuous verification system reveal that, while frog-boiling attacks are effective on fixed keystroke (password based) verification systems, they are not as effective on continuous keystroke based verification systems.

(6) Our results show that cognitive rhythm features can be used to predict a user's cognitive load level with varying degrees of accuracy, depending upon the granularity at which cognitive load level is predicted. Our experiments also demonstrate that cognitive rhythm features are good predictors of three demographic indicators: "sex", "handedness", and "Is user a native English speaker?"

All project deliverables, including software codes; feature catalogs; monthly reports; and copies of published and unpublished publications have been submitted through the DARPA T-FIMS system.

2 INTRODUCTION

2.1 Background and Motivation

The activities performed under this grant combine work on keystroke dynamics for user authentication with stylometric analysis that has been successfully applied to authorship studies. One of the major accomplishments of this effort is the development and subsequent validation of software codes and methods that are capable of extracting keystroke dynamics, stylometric features, and novel production features from typing data, to capture the cognitive and non-cognitive aspects of a user's text production behavior. Previous approaches to stylometry only operated on static text, and did not combine the linguistic information with language production (pausing, and revision). On the other hand, previous proposals in keystroke based authentication used a limited set of features. In comparison, we built users' profiles to capture a broad spectrum of new atomic, pausality, linguistic, and revision traits exhibited when a user produces text.

Another important difference between our effort and previous work in keystroke authentication is as follows: most studies in continuous authentication based on keystroke dynamics used data collected from users who typed text *copied* from a transcript. In a real authentication setting however, users typically compose free text (*e.g.*, the case of an email) than copy from an existing document. The former activity is likely to have a cognitive load different from that of the latter (due to the thought processes associated with text composition, production, and revision), and should, intuitively result in considerably different pause and revision behaviors for any given user. To ensure that our results depicted a realistic active authentication system, we opted against the "copy-typing" approach that has been adopted in the past studies.

Under this project, we conducted a suite of authentication experiments on data collected from 486 users who typed responses to a series of questions. Because users' typing behavior may depend on the "amount" of cognition required to produce a linguistic unit, we used questions having a wide range of cognitive loads. For example, we included questions involving: 1) retrieval of knowledge from long-term memory; 2) explaining, summarizing and interpreting of facts; 3) applying, executing, and implementing; 4) organizing and breaking material into constituent parts; 5) critiquing and making judgment based on criteria, and, 6) generating, planning and putting elements together to make a whole.

The methods and procedures for (1) data cleaning and processing; (2) availability and discriminability analysis, (3) authentication experiments, (4) non-zero effort attack experiments, and (5) experiments on prediction of cognitive load and demographic information, are detailed in Section 3 and the corresponding results and findings are detailed in Section 4. Next, we begin by providing details on performer sites and the project team.

2.2 Performer Sites and Team

2.2.1 Performer Sites

1. New York Institute of Technology (NYIT), Old Westbury, New York (Lead)
2. Louisiana Tech University, Ruston (LTU), Louisiana
3. Queens College, The City University of New York (CUNY), Queens, New York

2.2.2 Team

Investigators

1. Kiran Balagani (PI), NYIT
2. Vir Phoha (Co-PI), LTU
3. Andrew Rosenberg (Co-PI), CUNY

Student Researchers

1. Patrick Koch, NYIT
2. Sathya Govindarajan, NYIT
3. Raviteja Pokala, NYIT
4. Ming Xu, NYIT
5. Zibo Wang, LTU
6. Abdul Serwadda, LTU
7. Shafaeat Hossain, LTU
8. David Brizan, CUNY
9. Adam Goodkind, CUNY

2.2.3 Deliverables

2.2.3.1 Software Codes and Monthly Reports

As per the Schedule and Timeline in our Technical and Management Proposal (Volume 1), we uploaded software codes for feature extraction, authentication, and evaluation to the TFIMS system. We also uploaded all monthly reports to the TFIMS system.

2.2.3.2 Publications

Under this project, we published *1* archival journal paper, *1* magazine paper, and .*1* peer-reviewed workshop paper. Citations of *published* papers are:

1. Khandaker A. Rahman, Kiran S. Balagani, Vir V. Phoha, "Snoop-Forge-Replay Attacks on Continuous Verification With Keystrokes," *IEEE Transactions on Information Forensics and Security*, Vol.8, No.3, pp.528-541, March 2013.

2. Abdul Serwadda, Zibo Wang, Patrick Koch, Sathya Govindarajan, Raviteja Pokala, Adam Goodkind, David Guy Brizan, Andrew Rosenberg, Vir Phoha, Kiran Balagani, "Scan-based Evaluation of Continuous Keystroke Authentication Systems," *IEEE IT Professional*, Vol. 15, No. 20, pp. 20-23, July/August 2013.

3. Md. S. Hossain, Kiran S. Balagani, Vir V. Phoha, "On Controlling Genuine Reject Rate in Multi-stage Biometric Verification.," In Proceedings of the *IEEE Computer Society CVPR Workshop on Biometrics*, Portland, Oregon, June 2013.

3 METHODS, ASSUMPTIONS, AND PROCEDURES

3.1 Data Used in Active Authentication Experiments

For evaluation, we used data collected from 1013 subjects at Louisiana Tech University during April 2012 through June 2012. For each user, data was collected in two separate sessions and no two sessions happen on the same day. A total of 589 subjects participated in both Session 1 and Session 2. A total of 416 subjects participated in Session 1 only. A total of 8 subjects participated in Session 2 only. The data collection effort was ***NOT*** financed by the DARPA Active Authentication award. The data collection effort was financed by Co-PI Dr. Vir Phoha's various grants.

3.1.1 Subject Population Characteristics

- **Gender:** Male (569), female (427), information not available (17).
- **Ethnicity:** White (473), African (119), African American (174), Asian (200), Hispanic (20), White/African (1), Creole (1), Multi (2), Cuban (1), Aryan (1), Native American (1), Native (1), Indian (1), French (1), Guyanese (1), Other (40)
- **Age:** Minimum (17), Maximum (56), Average (21.608)

Majority of the subjects were college students, but also faculty and staff participated in the data collection activity. The counts are indicated in parenthesis.

3.1.2 Data Collection Procedure

Each participant composed and typed answers to 10 to 13 questions per session (see Appendix for a list of questions). Two different sets of questions were used for Session 1 and Session 2. Each question was chosen such that it belonged to one of the six cognitive load levels defined in [1]. Table 1 gives example questions and their corresponding cognitive load levels. For each question, the participant was asked to type an answer containing at least 300 characters. So, each participant produced a total of at least 3000 characters of text. A participant's typing session ended after he/she typed answers to all questions. The participants took approximately between 45 minutes and 2 hours to answers all questions. While typing the answers, participants were allowed to revise text by using Delete/Backspace keys. A keystroke sensor recorded the key code and timestamp of each key pressed and released. In addition, we collected the following data about a participant: a) typing experience, b) age, c) gender, d) right- or left-handed, e) native language, f) business language, and g) average number of hours a participant typed in a day.

Data collection apparatus consisted of a Dell desktop equipped with Windows XP OS, a QWERTY keyboard, and a mouse. We used GUI based software written in C# to collect data. The software provided text boxes for typing answers and transition buttons to switch between questions. A keylogger with 15.625 milliseconds clock resolution was used to record text and keystroke event timestamps.

Table 1. Cognitive Load Levels of Questions

Cognitive Load Level	Description	Example Questions
1	Involves retrieving knowledge from long-term memory	What is your favorite vacation spot? (1–Recall) Why do you like to visit there? (2—Explain)
2	Involves explaining, summarizing, and (or) interpreting	
3	Involves applying, executing, and (or) implementing	Discuss step-by-step instructions for accomplishing a task (e.g., mechanical tasks such as cooking a specific dish, building or repairing something, etc.).
4	Involves organizing and (or) breaking material into constituent parts	Give step-by-step driving directions to your favorite place in/around Ruston, from Louisiana Tech campus.
5	Involves critiquing and (or) making judgments based on criteria	What email do you use (e.g., Yahoo, Hotmail, Gmail, Webmail, etc.)? What improvements would you like to see in them?
6	Involves generating, planning, and (or) putting elements to together to make a whole	If you were to draw a picture of any type of landscape you wanted, what objects would you include in it? How would you go about drawing the landscape?

3.1.3 Data Cleaning

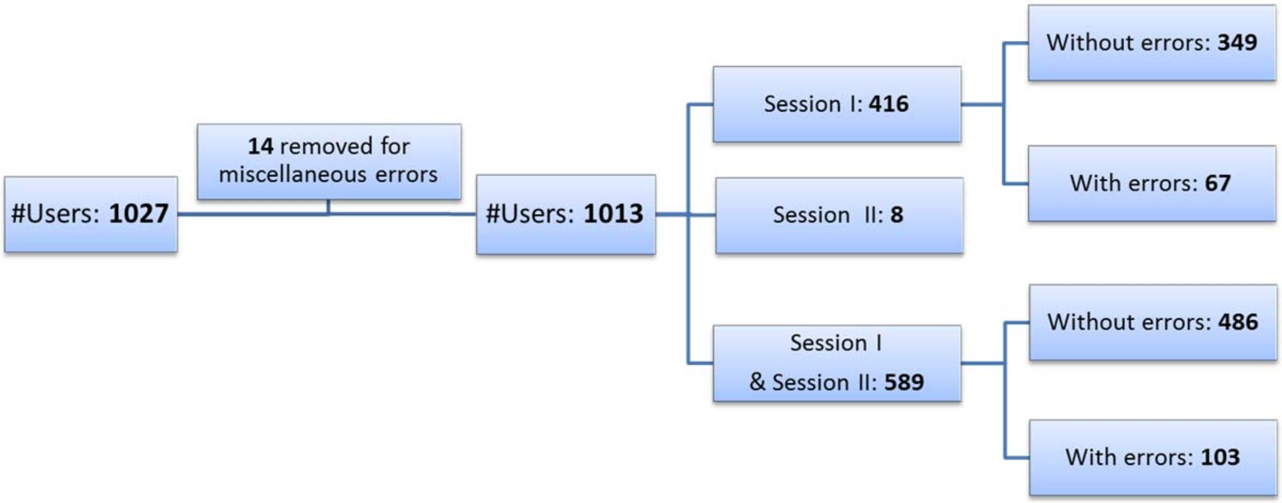


Figure 1. Subject Population Sizes Before and After Data Cleaning

Figure 1 shows the number of subjects finalized for our experiments after thoroughly inspecting the subjects' data for errors. We programmatically and manually inspected each subject's data for errors. Observed data errors primarily fell into five categories: (1) a subject with missing final text (software error), (2) a subject retyping questions instead of answers, (2) a subject typing gibberish or irrelevant text in one or more answers, (3) a subject repeating answers of previous questions, and (5) a subject for whom one or more answers were missing.

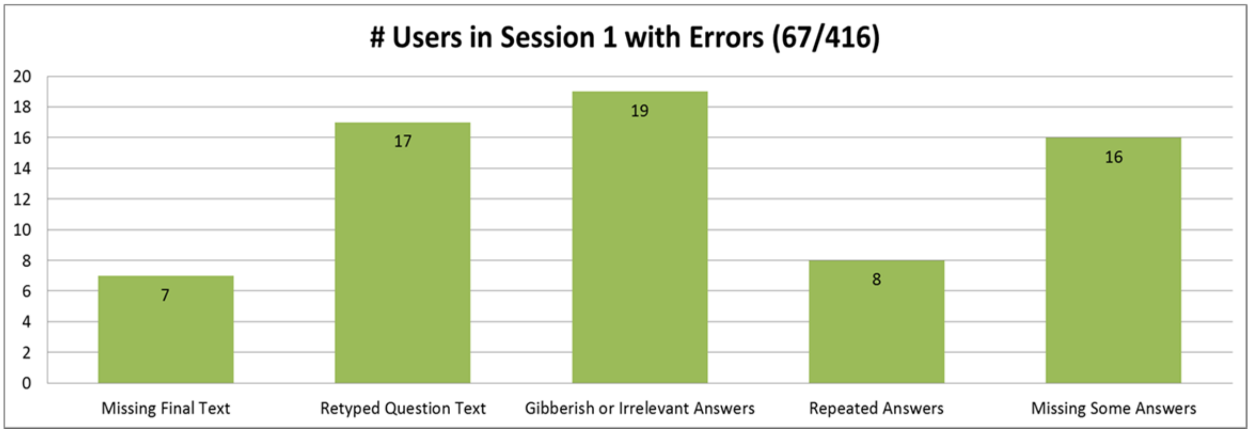


Figure 2. Number of Session 1 Subjects with Data Errors

Figure 2 gives the number (in y-axis) of subjects who participated in Session 1 and their corresponding errors (in x-axis). These subjects were not used in our experiments.

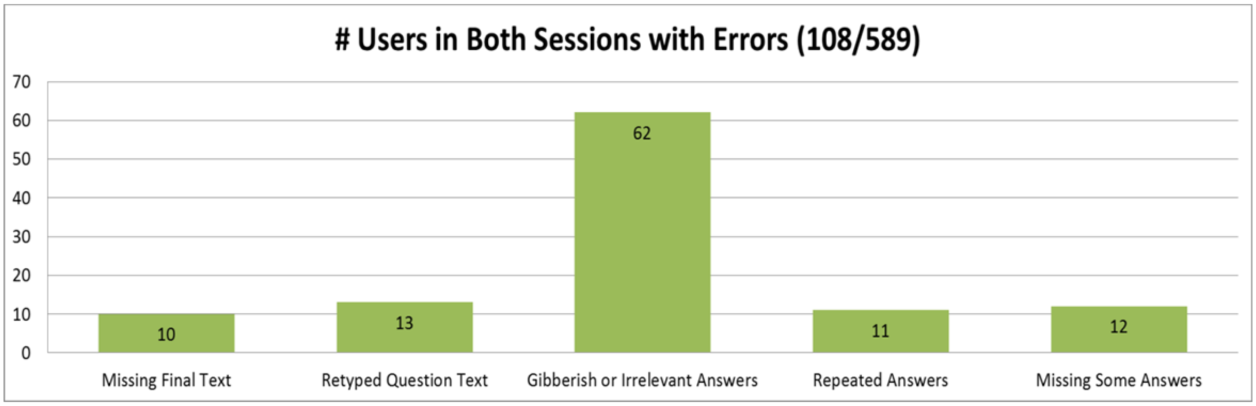


Figure 3. Number of Subjects who Participated in Sessions 1 and 2 with Data Errors

Figure 3 gives the number (in y-axis) of subjects who participated in Session 1 and Session 2 and their corresponding errors (in x-axis). These subjects were not used in our experiments.

3.2 Cognitive Rhythm Features

One of the most important outcomes of this effort was the development of software codes to extract atomic, pausality, and linguistic features from the typing data. The feature engineering effort resulted in a set of over 45 types of linguistically and cognitively motivated features, which resulted in more than 9000 individual features. The features describe a wide range of qualities related to the production of language capturing both physical and cognitive qualities. This effort represents a novel way to analyze information in keystroke data. The features were developed in a common codebase framework used by the performer sites. This common codebase enabled close collaboration between the sites. The extracted features are primarily divided into four types: (1) atomic, (2) pausality, (3) linguistic, and (4) revision. Below we briefly describe the four types of features.

Table 2. Atomic Features Extracted from the Typing Data

Feature	Description
Key hold latencies	Latency between press and release of a key
Key interval latencies	Latency between release of a key and the press of next key
Digraph latencies	Duration between press of the key and press of the second key in a digraph (“press-release-press-release”) sequence
Slur digraph latencies	Duration between press of the first key and press of the final key in the trigraph (anything other than press-release-press-release-press-release) sequence
Slur keys intervals	Latency between release of a key and the press of next key (here, press occurs before release)
Trigraph durations	Duration between press of the key and press of the final key in a trigraph (press-release-press-release-press-release) sequence
Slur trigraphs	Duration between press of the first key and press of the final key in the trigraph (anything other than press-release-press-release-press-release) sequence
Group keystrokes by "touch typing zone"	Touch typing designates a specific digit for each keyboard key. When looking at holds and intervals, replace the specific keystroke with the "zone" it is in. This will increase coverage.

Table 3. Pausality Features Extracted from the Typing Data

Feature	Description
Pause after long words	Calculates the pause length after words with more than 6 letters
Duration between pauses	Calculates the duration in milliseconds between successive pauses (> 250ms)
Keystrokes between	Calculates the duration in keystrokes between successive pauses (>250ms)

pauses	
Pause before period	Counts the number of pauses before a period
Pause after period	Counts the number of pauses after a period
Pause before comma	Counts the number of pauses before a comma
Pause after comma	Counts the number of pauses after a comma
Words between pauses	Counts the number of words begun between each pause; returns an array of counts
Characters between sentence beginning and first pause	Measures the number of characters between the beginning of the sentence and the first pause
Words between sentence beginning and first pause	Measures the number of words between the beginning of the sentence and the first pause
Pauses before modals	Counts the number of pauses before a modal
Pauses before nouns	Counts the number of pauses before a noun
Pauses before verbs	Counts the number of pauses before a verb
Pauses before modifiers	Counts the number of pauses before a modifier
PP bursts	Measure words, characters, time, major/minor delimiters, whitespace, word length etc. within pauses. Note: This is a refinement of what is done in "Duration Between Pauses".

We list the atomic and pausality features extracted under this project in Table 2 and Table 3. Most of the feature descriptions are self-explanatory. However, below we briefly describe the features for completeness.

Atomic features (Table 2) constitute key hold, key interval, digraph, and trigraph latencies and their "slur" counterparts in which the "key press–key release" sequences are reversed. We also extract timing between keys based on their keyboard position. Keys are grouped by row and canonical finger that would be used in "proper" touch-typing. These hold and intervals are normalized per user to measure, for example, if a typist is faster with keys on the right or left side of the keyboard.

To measure users' behavior as it relates to **pauses**, we developed a number of features to measure what a user does between two pauses, so called "pause bursts" (Table 3). We measure the number of words and characters between pauses. We also measure the number of words and characters used between the beginning of a sentence and the sentence's first pause. Further, we also measure the number of correctly spelled or misspelled words in each sentence and the pauses before or after each occurrence.

We measure **typing speed** in two ways: we look at the speed over the whole document and, following [2], the intra-word typing speed. We also look at timing ratios: for example, does a user take twice as long to produce a 2-letter word as she does to produce a 1-letter word?

To capture **typing timing**, we extract the traditional keystroke dynamic features of key hold and key interval. However, we extend these such that they are more reflective of the linguistic properties of the words, or characters that are being typed. These features can be viewed as an expansion of character-type identification. This measures user transition times between categories, e.g. a vowel to a consonant, or a vowel to another vowel.

We also extract the pausing behavior around punctuation, to measure how long a typist pauses before or after phrase-marking punctuation (e.g., periods, commas, colons).

Table 4. Linguistic Features Extracted from the Typing Data

Feature	Description
Number of sentences per answer	Breaks answer into complete sentences; method can discriminate between, e.g. "Mr.", and sentence-ending punctuation
Number of characters per answer	Calculates number of printable characters
Number of words per answer	Calculates number of words
Average word lengths (in chars)	Calculates average word length, in alphanumeric characters, per sentence
Sentence length, in words, per answer	Calculate average sentence length in words
Sentence length, in characters, per answer	Calculate average sentence length in characters
Type-token ratio	Calculates number of unique words divided by total number of words
Moving-average type-token ratio (MATTR)	Calculates type-token ratio for moving window of text; helps to control for length of text
Small-to-large window MATTR ratio	Calculates a ratio between a small window and large window MATTR; high ratios are indicative of internal repetition
Lexical Density	Ratio of lexical words (nouns, adverbs, adjectives, etc.) to total words
Nouns Per Sentence	Counts the number of nouns per answer
Verbs Per Sentence	Counts the number of verbs per answer
Modifiers Per Sentence	Counts the number of modifiers, e.g. adverbs and adjectives, per answer
Modals Per Verb	For each verb occurrence, counts how often it is qualified by a modal, e.g could, should, etc.
Named Entity Rate	Measures the average number of Named Entities (recognizable people, places and things) per sentence
Semantic Graph Edge Count	Counts the edges in a semantic graph

N-letter word containing K backspaces	Counts the no. of backspaces in N-letter words.(N = 1 - 21)
Average TF*IDF	Calculates tf*idf (term frequency * inverse document frequency), and then averages all scores across an answer

We developed code modules to count linguistic qualities present in an answer (Table 4). We count words, sentences, characters, keystrokes, syntactic phrases and semantic units. The module to count sentences takes advantage of the open source platform OpenNLP, who's Sentence Terminator Detection API can differentiate between sentence-ending punctuation, and punctuation such as the period in "Mr. Smith." We also built modules to count the number of printable characters, as well as the number of words. We then built a number of features which measure average word length and average sentence length (in both words and characters).

We then designed a number of features to measure **lexical diversity**, and the number of different words used in an answer. We designed a feature to measure the type-token ratio, or the number of unique words divided by the total number of words. A similar feature was designed to measure the different number of part-of-speech tags divided by total words, a feature commonly known as "l**exical density**".

Because type-token ratio, though, is sensitive to the size of a document, i.e. it tends to get lower as a document gets longer [3], we also implemented a feature to measure Moving-Average Type-Token Ratio (MATTR). This metric is not sensitive to length, at all [3]. When measuring MATTR, it is important to pick an accurate window size for the moving average. Also, comparing two window sizes, a small one and a large one, provides information about internal repetition within a text [3]. We implemented a feature to measure these window ratios on user's answers.

We also measured the Inverse Document Frequency (IDF) of each word in an answer. IDF measures the uniqueness of each word, as it is expected that users' typing will be different for highly unique words, versus common words.

To measure **lexical** complexity, we also examine the ratio of 1-letter, 2-letter, 3-letter, etc. words to the overall number of words. Extending this idea down to the character level, we extract character-type ratios. We check if a character is of a specific type, e.g., alphabetical, uppercase, etc. Once the totals of each character-type are compiled, a ratio is output.

Because a user's typing will also depend on the **parts-of-speech** used, we developed features which can measure the number of nouns, verbs, modifiers, and modal verbs per sentence. Further, users often uniquely change typing patterns when typing highly recognizable names and places, called Named Entities, e.g. George Washington or Mississippi. As such, we developed a module which can pick out, and measure, common NEs, using Stanford's Named Entity Recognizer.

We also developed features to measure the basic size and shape of a **semantic graph**. We measure both the number of edges in the graph as well as the number of edges per word. We measured the average depth of a semantic graph. Further semantic graph features will be developed to take advantage of the rich data available in a sentence's semantic graph.

We expect users to behave differently when typing different types of **punctuation**. As such, we developed features to measure the average pause length both before and after common types of punctuation, such as periods and commas. Further, we also measure the average length of a pause both before and after each part-of-speech.

Table 5. Revision Features Extracted from the Typing Data

Features	Description
On leading edge features	Measures the time the user is writing new words
Away from leading edge features	Measure the time the user has spent revising.

Our **revision features** (Table 5) measure how a subject behaves when they are "in a revision," or not at the leading edge of their typing session. We currently count the time spent in revision, and the number of characters produced in a revision. These are reported as both absolute figures, as well ratios of overall figures.

We also identify pause and revision behavior before and after **misspelled** words. Recognizing misspelling in static text is trivial. Words like "teh" and "becuase" and "tjerefpre" do not exist in a dictionary of English words and can be considered misspelled. However, misspellings are often corrected by a user before the answer is completed. Identifying misspelling in keystroke data requires us to recognize that the sequence "H, O, W, V, *BKSP*, E, V, E, R" includes a misspelling. The approach we take is to recognize that "HOWV" is not a valid prefix to any English word. Regardless of what the typist does after typing 'V' in this sequence, we know that this is a misspelled word. To accomplish this we use a 'trie' data structure. This is a prefix tree which allows for rapid "O(1)" querying of whether a keystroke indicates a misspelling or not. Using this information we calculate the revision rate, typing speed, and pause behavior relative to misspelled versus correctly spelled words. We also measure how frequently words are misspelled.

3.3 Availability and Discriminability Analysis

Having discovered over 9000 features, it is important to select important features out of this huge set that represent most of the data and that are highly reliable for achieving good discrimination.

Note: In Table 2, Table 3, Table 4, and Table 5 we only specify different *types* of features extracted under this effort. However, the actual number of individual features is over 9000. For example, we extracted 97 different key hold latencies, over 8000 different key interval and digraph latencies, and so on.

We measure availability using *Hit Ratio* (HR) and discriminability using *Symmetric Uncertainty* (SU) measure [4]. Hit Ratio is defined as follows:

$$Hit\ Ratio = \frac{\#\ of\ scans\ with\ feature\ F}{Total\ \#\ of\ scans} \tag{1}$$

For a specified scan length (or detection time), hit ratio of a feature *F* is the measure of number of recorded values of that feature among the total number of recordings or scans.

Symmetric Uncertainty is defined as follows:

$$Symmetric\ Uncertainity = \frac{2\ \times I(User; Feature)}{H(User) + H(Feature)} \tag{2}$$

Symmetric Uncertainty is a measure of goodness of features for classification.

We performed discriminability and availability analysis on the data from 349 users who participated in the Session 1 only. Additionally, for the purpose of discriminability and availability analysis, we set the scan duration of each authentication attempt to 1, 5, and 10 minutes.

3.4 Authentication Methods and Experiment Design

Here, we discuss: (1) the authentication methods used in our project; (2) how we designed the verification (1-to-1 matching) and identification (1-to-N matching) experiments; and (3) the definitions of error metrics used to report our results.

3.4.1 Authentication Methods

3.4.1.1 Scaled Manhattan Distance Authenticator

In this authenticator, the distance between template and a testing attempt is calculated as:

$$\text{Scaled Manhattan Distance} = \frac{1}{M}\left[\sum_{i=1}^{M}\frac{|x_i - y_i|}{\sigma_{x_i}}\right], \quad (3)$$

where M is number of matched features, x_i is the i_{th} feature in the template and y_i is the corresponding feature in the testing attempt. σ_{x_i} is the standard deviation of x_i.

3.4.1.2 Scaled Euclidean Distance Authenticator

In this authenticator, the distance between template and a testing attempt is calculated as:

$$\text{Scaled Euclidean Distance} = \frac{1}{M}\sqrt{\sum_{i=1}^{M}\frac{(x_i - y_i)^2}{\sigma_{x_i}}}, \quad (4)$$

where M is number of matched features, x_i is the i_{th} feature in the template and y_i is the corresponding feature in the testing attempt. σ_{x_i} is the standard deviation of x_i.

3.4.1.3 Relative Distance Measure Authenticator

"Relative" measure or "R" measure was proposed by [5]. It measures the degree of disorder between two arrays of same length. It is similar to Manhattan distance, except that the distance is calculated based on the relative positions of same elements or features in the two arrays with respect to the maximum disorder of the arrays. The maximum disorder occurs when two arrays have elements in reverse order. The maximum disorder of an array A is computed as:

$$Maximum\ Disorder\ (MD) = \begin{cases} \frac{|A|^2}{2}\ if\ |A|\ is\ even \\ \frac{|A|^2 - 1}{2} if\ |A|\ is\ odd \end{cases} \quad (5)$$

where $|A|$ represents the length of the array. Finally we compute R-Measure as follows:

$$\text{R Distance} = \frac{1}{MD}\sum_{i=1}^{M}\left|Pos_x(f_i) - Pos_y(f_i)\right|, \tag{6}$$

where $Pos_x(f_i)$ and $Pos_y(f_i)$ are positions of feature f_i in sorted feature instance arrays of template x and test y. MD is the maximum disorder between x and y.

3.4.1.4 Score-level Fusion with Weighted Sum Rule

We implemented score-level fusion with weighted-sum rule. Let $\{A_1, A_2, \cdots, A_i\}$ denote i individual authentication algorithms. Let $\{FS_1, FS_2, \cdots, FS_j\}$ denote j different feature sets from the template. For example, FS_1 can be a set of pause features, FS_2 a set of revision features, FS_3 a set of key press latencies, and so on. Let $\{(A_1, FS_1), (A_2, FS_2), \cdots, (A_i, FS_j)\}$ denote a set of authenticator feature-set pairs. For example, $(A_{i\,,}FS_j)$ represents an authenticator A_i using features in FS_j to generate authentication scores. Let $S_{11}, S_{12}, \dots, S_{ij}$ represent the output scores generated by $(A_1, FS_1), (A_2, FS_2), \dots (A_i, FS_j)$ respectively. In weighted sum fusion, the fused score S is computed as, $\omega_{11}S_{11} + \omega_{12}S_{12} + \cdots + \omega_{jk}S_{jk}$, where $\{\omega_{11}, \omega_{12}, \cdots, \omega_{jk}\}$ are weights subject to the constraint $\omega_{11} + \omega_{12} + \cdots + \omega_{jk} = 1$. Though the rule is simple, it has been shown to be surprisingly effective in biometric authentication problems and is widely considered (see [6]-[9]) as a benchmark for evaluating fusion algorithms.

3.4.2 Verification Experiment Design

For verification experiments, we used 486 subjects who participated in Session 1 and Session 2 during the data collection. Session 1 typing data was used for training (to create user templates) and Session 2 data used was used to create testing attempts. We experimented with testing attempts of different scan lengths, ranging between 2 and 10 minutes. (In this document, scan length and detection time have the same meaning.)

To create genuine scores, a template of the i^{th} user was matched with test attempts of the same i^{th} user. To create zero-effort impostor scores, a template of the i^{th} user was matched with test attempts of the remaining 485 users other than the i^{th} user.

3.4.3 Identification Experiments

For identification experiments, we used 300 subjects who participated in Session 1 to create a gallery of registered subjects. To create the “registered-member” probe set, we used Session 2 data from 186 users (-because these users are present in the gallery, they are “registered-members”). To create the “non-registered member” probe set, we used Session 2 data from 186 non-member users (-because these users are not in the gallery, they are “non-registered members”). We experimented with testing attempts of different scan lengths, ranging between 5 minutes and 10 minutes.

3.4.4 Authentication Metrics

Here, we define the authentication metrics used to quantify the results of our experiments.

3.4.4.1 Verification Metrics

EER (Equal Error Rate): The crossover point where the false accept rate (FAR) (also called impostor pass rate) curve meets with the false reject rate (FRR) curve. We calculated FAR, FRR, and EER for the entire population (i.e., scores from all subjects), NOT for individual subjects.

3.4.4.2 Identification Metrics

FNIR (False Negative Identification Rate):

$$FNIR = \frac{Number\ of\ registered\ user\ attempts\ rejected\ by\ the\ system}{Total\ number\ of\ registered\ user\ attempts\ presented\ to\ the\ system} \quad (7)$$

Explanation: "Registered-user attempts rejected by the system" means the proportion of probes from the registered-member set that was erroneously rejected by the identification system. A probe is rejected if the correct user has not appeared among the "top 5" best scores. This is also called as "Rank 5" identification.

FPIR (False Positive Identification Rate):

$$FPIR = \frac{Number\ of\ unregistered\ user\ attempts\ accepted\ by\ the\ system}{Total\ number\ of\ unregistered\ user\ attempts\ presented\ to\ the\ system} \quad (8)$$

Explanation: "Unregistered-user attempts accepted by the system" means the number of probes from the nonregistered-member set that was erroneously accepted by the identification system. If any probe from the nonregistered-member set returns a non-zero number of candidate users, then we count it as an error.

3.4.4.3 Detection Time (Scan Time or Scan Length)

The verification and identification errors were reported for detection times between 2 and 10 minutes. We used detection time, which is the time period the system waits to collect the biometric test (authentication) signal for subsequent authentication. In this document, detection time, scan length, or scan time, all have the same meaning.

3.5 Snoop-Forge-Replay Attack Procedure

A detailed description of the snoop-forge-replay attack, attack parameters and configurations, methods, and results can be found in our published journal paper [10]. Here, we briefly discuss the snoop-forge-replay attacks for completeness.

We developed a new attack called the snoop-forge-replay attack on keystroke based continuous verification systems. The attack is not specific to any particular keystroke based continuous verification method or system. It can be launched with easily available keyloggers and APIs for keystroke synthesis. Our results from 2640 experiments show that: (1) the snoop-forge-replay attacks achieve alarmingly high error rates compared to zero-effort impostor attacks, which have been the de facto standard for evaluating keystroke based continuous verification systems; (2) four state-of-the-art verification methods, three types of keystroke latencies, and eleven matching-pair settings (-a key parameter in continuous verification with keystrokes) that we examined were susceptible to the attack; (3) the attack is effective even when as low as 20 to 100 keystrokes were snooped to create forgeries. In light of our results, we question the security offered by current keystroke based continuous verification systems.

3.5.1 Snoop-Forge-Replay Attack Flowchart and Description

Here, we briefly discuss the procedure for launching a snoop forge replay attack.

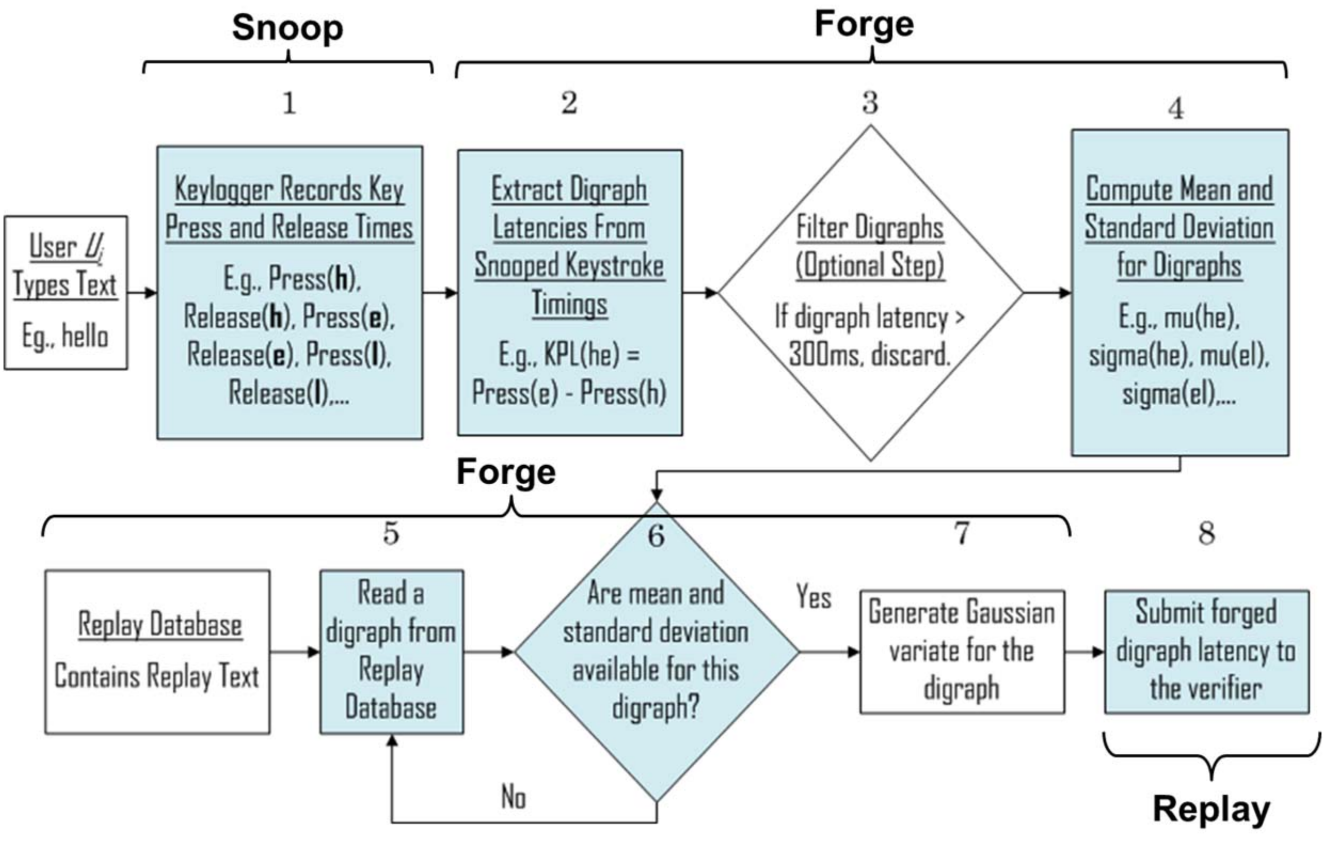

Snoop
Forge
1
2
3
4
User U_j Types Text
Eg., hello
Keylogger Records Key Press and Release Times
E.g., Press(h), Release(h), Press(e), Release(e), Press(l), Release(l),...
Extract Digraph Latencies From Snooped Keystroke Timings
E.g., KPL(he) = Press(e) - Press(h)
Filter Digraphs (Optional Step)
If digraph latency > 300ms, discard.
Compute Mean and Standard Deviation for Digraphs
E.g., mu(he), sigma(he), mu(el), sigma(el),...
Forge
5
6
7
8
Replay Database
Contains Replay Text
Read a digraph from Replay Database
Are mean and standard deviation available for this digraph?
Yes
No
Generate Gaussian variate for the digraph
Submit forged digraph latency to the verifier
Replay

Figure 4. Snoop-Forge-Replay Attack Flowchart. Step 1 snoops keystroke timings. Steps (2)-(8) create and replay a forged verification attempt.

Figure 4 shows the flowchart of the attack. The attack is executed in three steps: 1) *snoop* (steal) a victim user's keystroke timing information using a keylogger, 2) *forge* a typing sample using the keystroke timing information stolen from the victim user, and 3) *replay* the forged typing sample in such a way that the continuous verification system thinks that it is the victim user who is typing. The goal of the attack is to submit forged typing samples to the verifier so that an attacker can access the computer without being detected. Below, we explain the steps in detail.

Snoop: In this step, the attacker steals a victim's keystroke timing information. For example, if the victim typed the text “this is snooped text”, the attacker records a series of timestamps corresponding to the presses and releases of the keys in the text. An attacker can snoop a victim's keystroke timing information using a hardware keylogger or a software keylogger. Software keyloggers have become the most popular forms of keyloggers because they can be easily developed, are easily available, and can be deployed from remote locations onto a victim's machine (e.g., using trojans and spyware).

Forge: In this step, we create a keystroke forgery of a victim user U_i at sample-level. A forgery has two parts: 1) “dummy” text and 2) a series of latencies between the press and release of letters in the dummy text. For example, a forgery of U_i can have the dummy text “**this is dummy text**”. The key hold and interval values for replay this text come from the snooped keystroke latencies of U_i.

Preparing “dummy text” file: The “dummy text” file supplies text to create a forgery. Technically, the file can contain any text, ranging from multiple repetitions of a single letter (e.g., aaa …) to a large text corpus representative of English language usage (e.g., Corpus of Contemporary American English (COCA) [www.americancorpus.org]). For our experiments, we created a dummy text file from with 497,184 words from COCA [10]. In addition, we added text from 20 Wikipedia documents.

Replay Using Keystroke Emulator: We developed a keystroke emulator that injects synthetic key press and release events. We programmed the emulator in Visual C++ and used SendInput API. The goal of the emulator is to use the snooped latencies to inject key press and release events for the dummy text in a way that the verifier thinks that it is the victim Ui who is typing the dummy text. An emulation algorithm gives the steps to forge and replay a victim user Ui's typing pattern. The input to the algorithm is a dummy text file and a series of key hold and interval latencies computed from Ui's snooped keystrokes.

3.5.2 A Brief Description of Experiments

We conducted 2640 attack experiments with 24 attack configurations, 10 individual and fusion verifier configurations, 11 matching-pair settings (24 x 10 x 11 = 2640). We analyzed the effect of four attack parameters: (1) number of snooped keystrokes (we experimented with 20, 50, 100, 200, 600, and 1200 snooped keystrokes; (2) filtering outliers in the snooped keystrokes (we experimented with and without filtering outliers; (3) Gaussian perturbation of snooped latencies (we experimented with and without perturbing latencies), and (4) frequency of occurrence of digraphs in snooped text (we experimented with 1, 2, and 3 occurrences of digraphs).

To generate sufficient number of snoop-forge-replay attacks for evaluation, we emulated the typing activity of a victim user for 24 hours (i.e., we executed a keystroke emulation program for 24 hours to generate sufficient number of forgeries for each victim). Because we experimented with 150 victim users and 24 attack configurations, we would have to run the emulator for 150 (victims) x 24 (attack configurations) x 24 hours = 3600 days (or approximately 10 years). To perform emulation at this scale, we set up a virtualization environment with 150 virtual machines. We dedicated one virtual machine for emulating a victim. For each attack configuration, we ran 150 emulators parallely on 150 virtual machines and reduced the emulation time to just 24 days. By parallely running 150 virtual machines, in 24 hours, we forged thousands of attacks against 150 users.

3.6 Frog-boiling Attack Procedure

Where keystroke dynamics is used as a second layer of defense to a short password string, it was recently shown that a systematic template morphing attack (i.e., frog-boiling attack [12]) can significantly degrade the performance of a keystroke authentication system. We evaluated the impact of a similar attack on continuous keystroke authentication.

3.6.1 Assumptions and Mechanisms

We simulated the frog-boiling attack based on the following assumptions:

1. An attacker is able to snoop on a user's typing session with the aid of a key logger.
2. The keystroke updating system uses the sliding window approach [13] to update user templates after each successful authentication attempt. In this particular updating approach, the oldest typing sample is replaced by the latest typing sample whenever authentication is carried out successfully.
3. While the frog-boiling attack is drifts a user template, the genuine user is still attempting to use the system. Hence as the attack drifts the user's template away from its true form, the user's successful authentication attempts drift it back towards its true form. The overall drift depends on the rate at which the user/attacker logs onto the system.

The mechanism of the frog-boiling attack in continuous keystroke authentication is similar to the design in fixed text authentication (see [12]). A frog-boiling attacker intends to drift a user template *S* towards a destination template *D*, where *D* represents the behavior from a population (i.e. population template). This drift is carried out in several short steps (*N* steps in total), each of which is governed by the equation:

$$F' = \begin{cases} x \cdot (F + (i-1) \cdot \Delta), & 1 \le i \le N \\ x \cdot (F + N \cdot \Delta), & i > N \end{cases} \tag{9}$$

The above equation generalizes the feature vector F', used to create the i^{th} frog-boiling authentication attempt. F represents the feature vector formulated from the victim's snooped typing samples. $\Delta = (D - F)/N$ is the drift size of each step of frog-boiling attack. The noise term x is modeled as a Gaussian random variable given by a mean 1 and standard deviation σ/μ, where μ and σ are the mean and standard deviation of the specified feature in snooped data.

3.6.2 Experiment Design

We used the first session of the data for building user templates. Session 2 was to make the baseline test. Phase 2 data (-collected during October-November 2012 in exactly the same manner as the dataset used in authentication experiments-) was used to create genuine attempts. User templates were created using 107 most frequent key hold and interval features as observed in the training set. We chose 100 users whose typing sessions in the training set lasted at least 20 minutes. The user template size was fixed to 20 minutes for each user. Authentication attempts were made based on 1-minute segments of the full 20-minute block. If an attempt passes the authentication, it replaces the oldest 1-minute scan in the template.

The attacker snoops 5 minutes of keystroke typing from a genuine user (see Assumption 1). This 5-minute sample is then used as a basis to morph the victim's template in several short steps (Recall the vector F in the above equation). The noise term x is modeled as a Gaussian random variable having a mean and standard deviation of the specified feature in snooped data.

In one set of experiments, we assume that both the attack and genuine samples arrive into the system according to a Poisson process with rates λ_A and λ_G respectively. In the other set, we assume that both the attack and genuine attempts arrive into the system at times following a uniform distribution. For either distribution, the ratio of the number of genuine (legitimate) to attack attempts is 1:1, 1:2, and 2:1 in our experiments.

3.7 Prediction of Cognitive Loads and Demographic Information

Our goal in this task was to (1) predict cognitive load (Level 1 through Level 6) of each typing instance, given the atomic, pausality, linguistic, and revision features and (2) to predict demographic information (handedness, gender, native English speaker or not). To predict cognitive loads and demographic information, we use the classifiers available in WEKA machine learning toolkit. Users were divided into train and test groups. No user was present in both the train and test set. Moreover, there are two sets of prompts at each cognitive load level. No prompt used in the training data was used in the test data. This guarantees that all of the experiments described in this section are user-independent and prompt-independent.

We predicted cognitive loads at a variety of different levels. The prompts were labeled from 1 through 6, where 1 asks a user to "Retrieve knowledge from long-term memory" and 6 asks a user to "Generate, plan and/or put elements together". The specific prompts (questions) are in Appendix. We perform a 6-way classification; we also group 1&2, 3&4, and 5&6 leading to a courser, 3-way classification. Finally, we explore 3 groupings of binary "high" vs. "low" classifications, 1 vs. 6, 1&2 vs. 5&6 and 1&2&3 vs. 4&5&6.

The classification approaches for handedness (Right vs. Left), gender (Male vs. Female) and native language (English vs. Non-English) were all binary classifications based on self-reports by the users. In these experiments, the number of users occasionally varies. Some users did not respond to one or more of these questions. In these cases their answer was omitted from analysis.

In each experiment, we explored Naïve Bayes, AdaBoost, J48, and SVM classifiers. All experimental results were reported only on the test data.

While the cognitive load data is approximately balanced by virtue of the data collection, the demographic indicators have a significant amount of skew. For example, ~90% of subjects are right handed. To encourage prediction of minority classes (i.e. left-handed, non-native and female) we used *undersampling* to generate a class balanced training data set. This involves discarding enough majority class data points such that the majority and minority classes are equally represented.

Note: We **do not** modify this distribution in the test data. We believe that this demographic skew is approximately representative of the world in which a deployed system would exist. We do not artificially modify the test data to make the classification easier, but rather do our best to modify the training data to get the most robust classifier possible.

3.8 Other Research Off-shoots of the Project

As a part of this project, we also investigated theoretical properties of *biometric verification with reject option* and validated them with NIST BSSR1 multimodal fusion dataset. We also investigated privacy-preserving protocols for keystroke based continuous authentication. Because these topics are off-shoots of the research performed under this project and are not the primary focus, we do not include further details.

4 RESULTS AND DISCUSSIONS

4.1 Results of Availability and Discriminability Analysis

In this section, we report the availability and discriminability of over 9000 key hold, interval (also same as digraph), trigraph features and there slur counterparts using hit ratio and symmetric uncertainty measures.

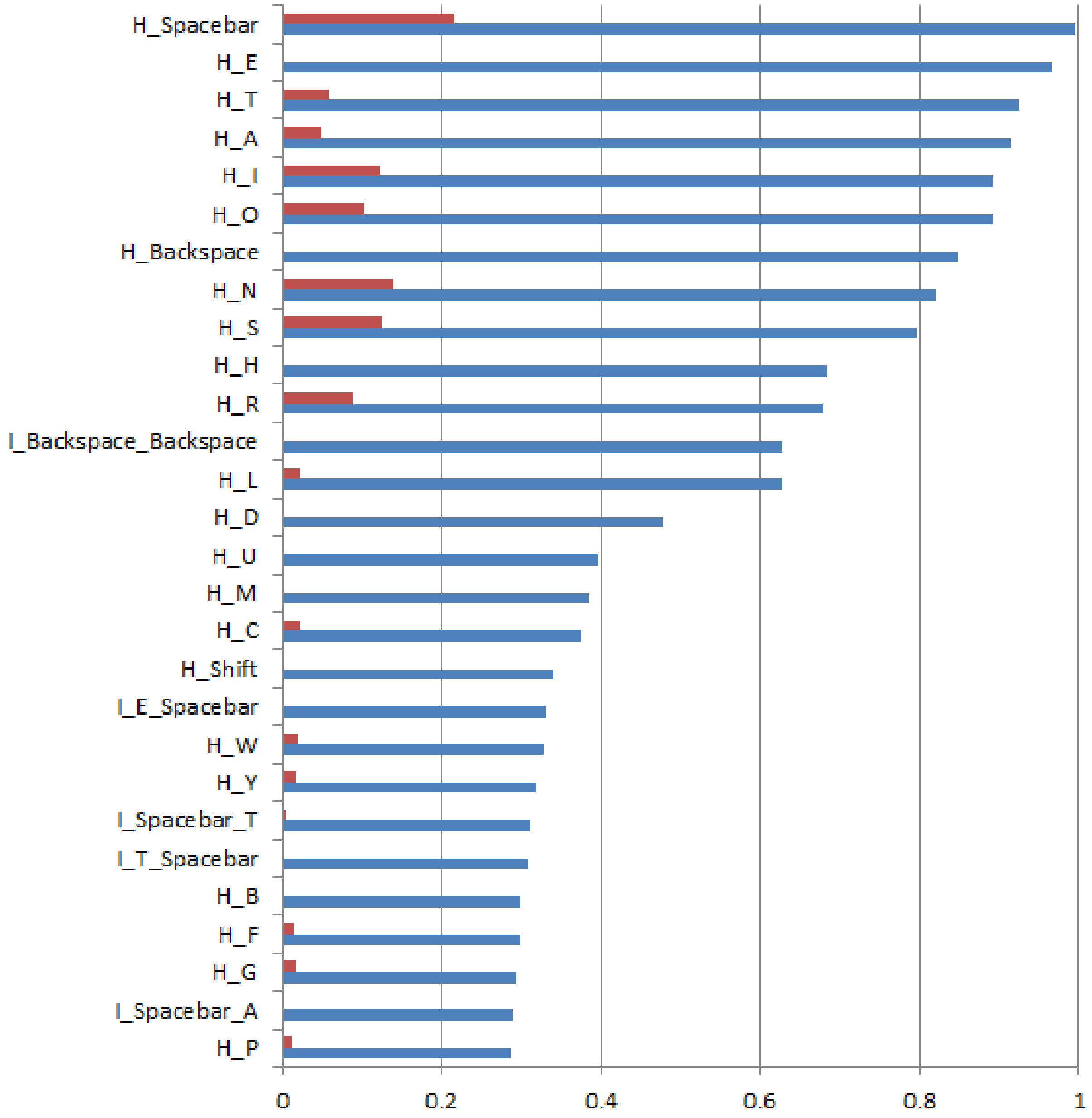


Figure 5. Hit Ratio (Blue Bars) and Symmetric Uncertainty (Red Bars) of Features in 1-minute Scans. There are 28 Key Hold and Interval Features out of Over 9000 Features

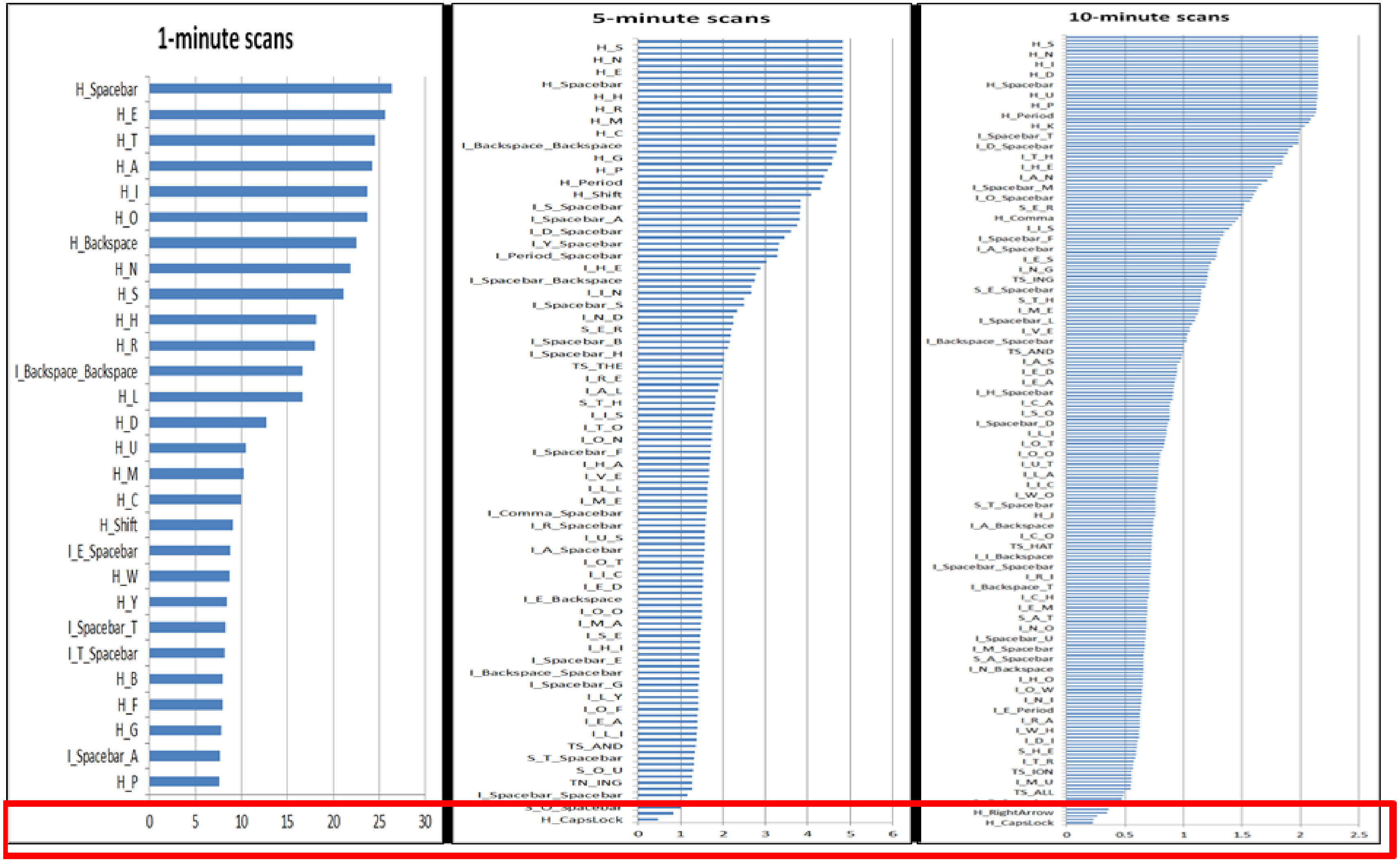


Figure 6. Comparison of Hit ratios of Key Hold Features for 1-, 5- and 10-minute Scans. 28 Key Hold and Interval Features in 1-minute Scans were Available 5 or More Times per User. Similarly, in 5-minute scans, the Number was 4 Times or More per User. In 10-minute scans, 28 Key hold and Interval Features were Available 2 or More Times per User

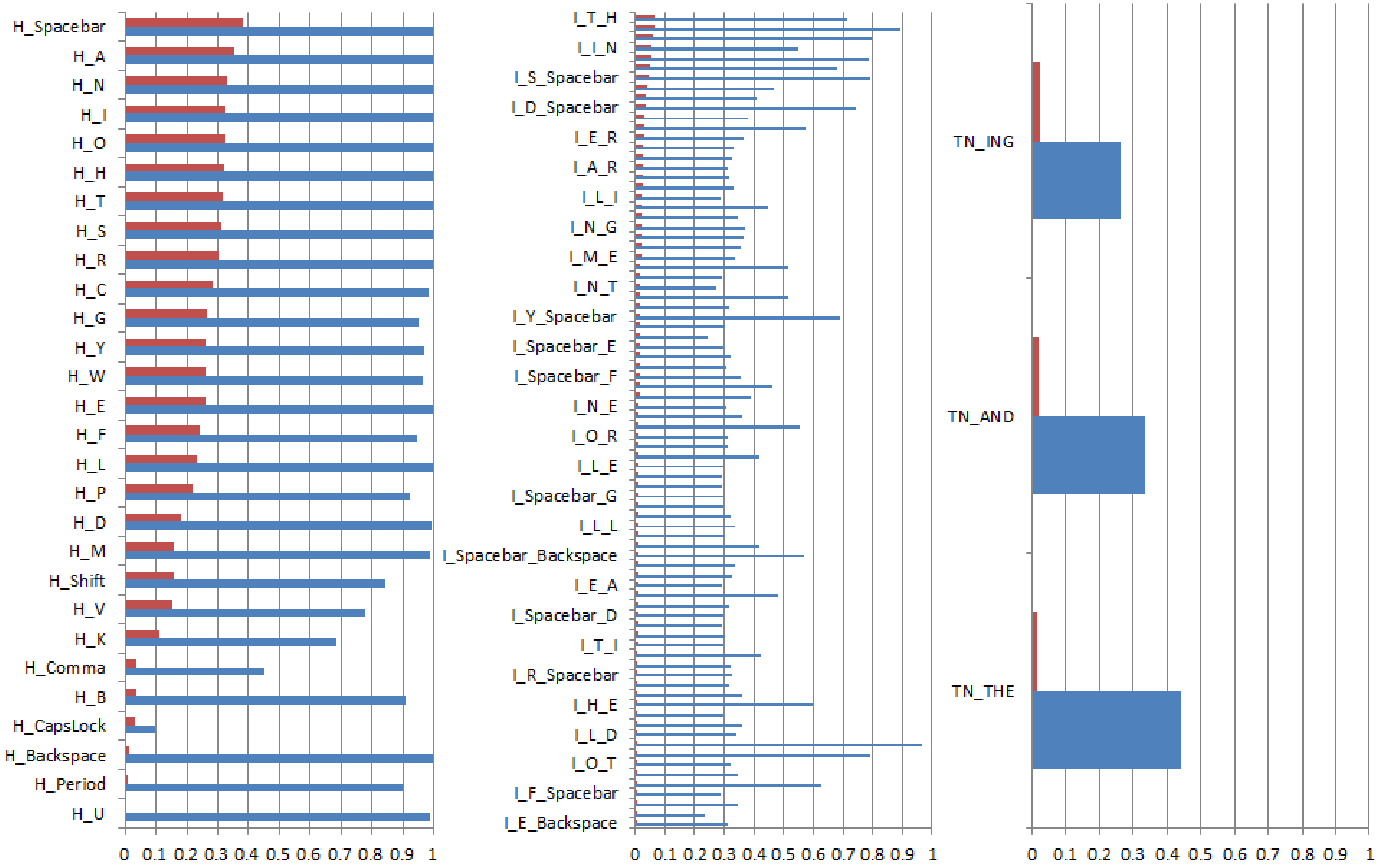


Figure 7. Hit Ratio and Symmetric Uncertainty of Key Hold, Key Interval, and Trigraph Features in 5-minute Scans

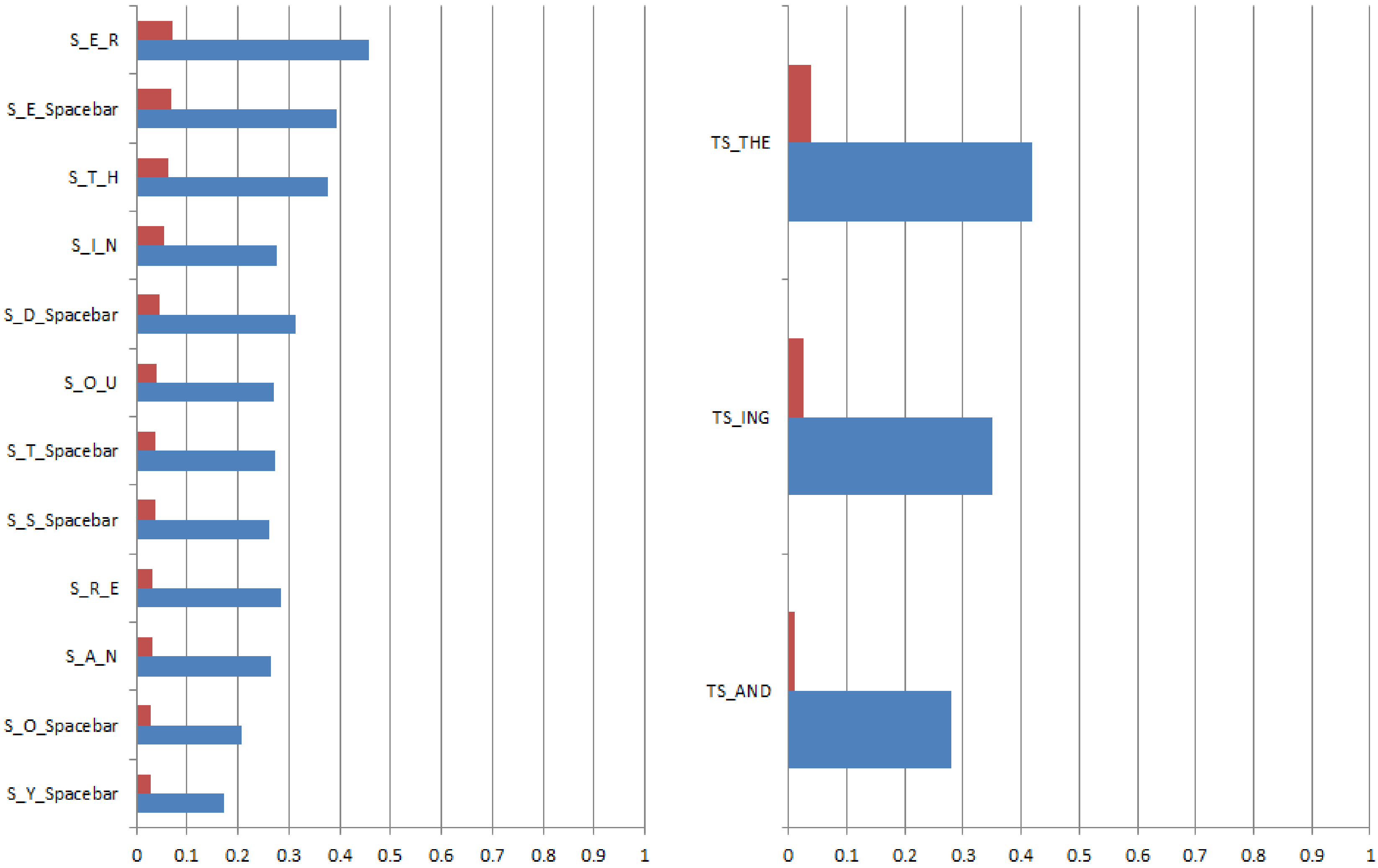


Figure 8. Hit Ratio and Symmetric Uncertainty for Slur Features in 5-minute Scan. Overall, 15 Slur Features were Observed

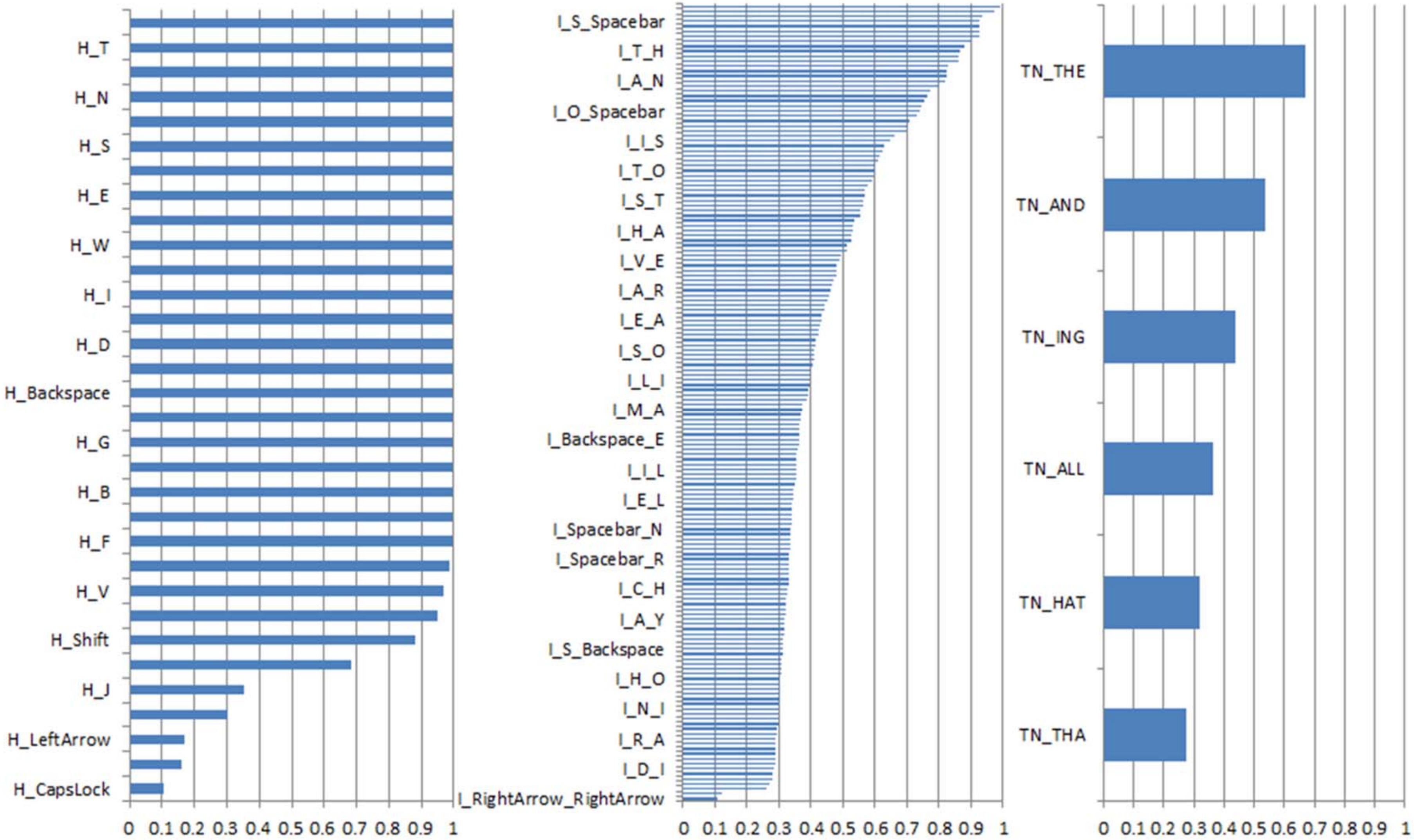


Figure 9. Hit Ratios of Features in 32 Key Hold, 160 Key Interval, and 6 Trigraph Features in 10-minute Scans

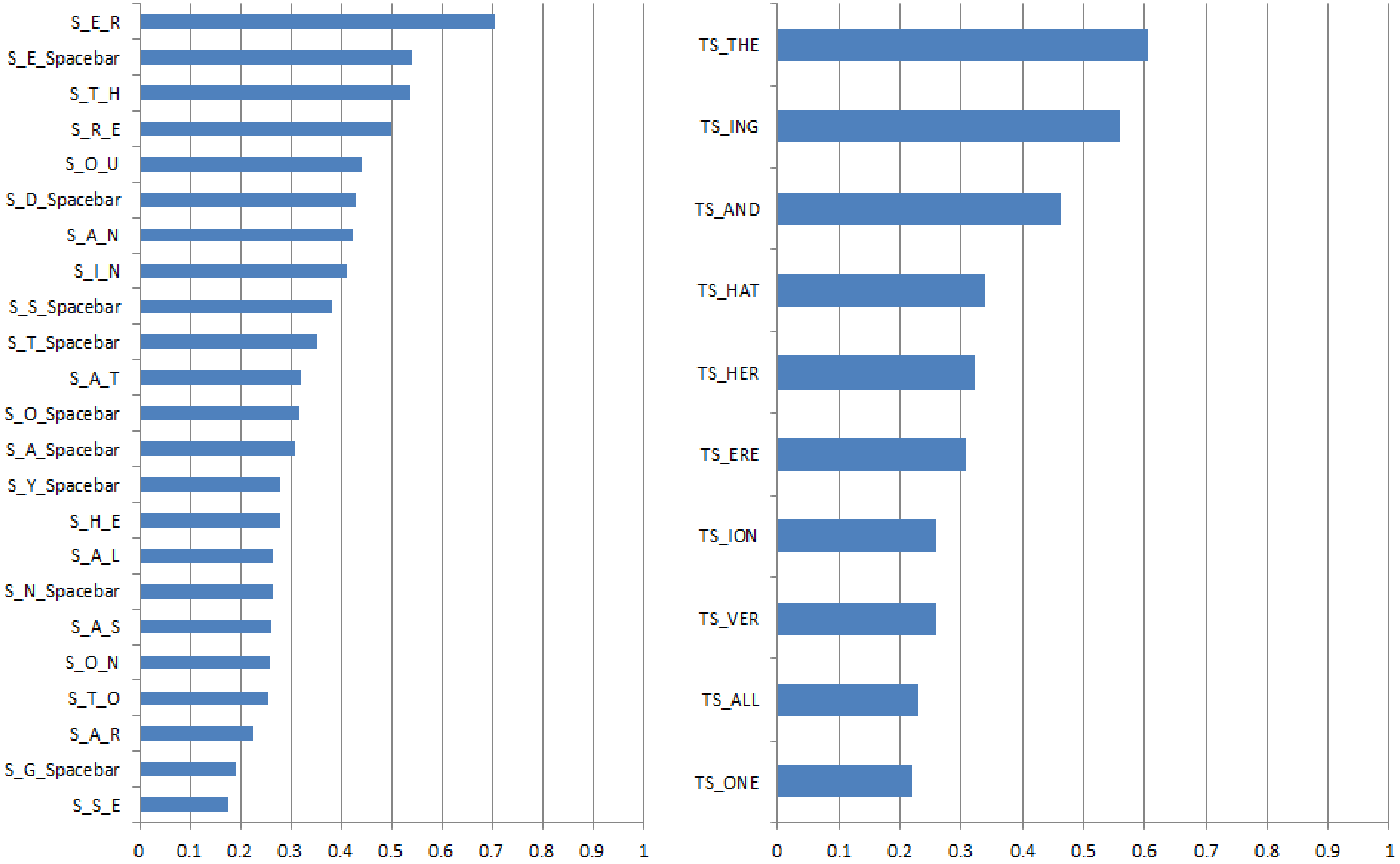


Figure 10. Hit Ratios of 23 Key Interval and 10 Trigraph Slur Features in 10-minute Scans

Approved for Public Release; Distribution Unlimited.

Discussion:

Figure 5 shows that for 1-minute scans, 28 key hold and key interval features were available. Overall, the key hold features had higher hit ratios and symmetric uncertainty than key interval features. However, not all key hold latencies with high hit ratio had high symmetric uncertainty, indicating that a high feature with high availability may not offer much discriminability.

In Figure 6, we compare the availability of features in 1-, 5-, and 10-minute scans. The figure shows that key hold and key interval features, overall, have higher availability than trigraph and "slur" features. For 1-, 5-, and 10-minute scans, the top 10 most available features were key hold features predominantly.

In Figure 7, we show the key hold, key interval, and trigaph features for 5-minute scans. Again, key hold features had higher overall hit ratios and symmetric uncertainty than key interval features. Trigraph latencies had the lowest hit ratios and symmetric uncertainty.

Figure 8 shows that the hit ratios and symmetric uncertainty for "slur" key interval and "slur" trigraph features were very low in 5-minute scans.

In Figure 9, we show the hit ratios of key hold, key interval, and trigraph features for 10-minute scans. Again, key hold features had the highest hit ratios followed by key interval and trigraph latencies. We did not calculate the symmetric uncertainty for features in 10-minute scans because the available sample size of each feature per user was not sufficient to determine the mutual information between a feature and the user.

In Figure 10, we show the hit ratios of 23 key interval and 10 trigraph slur features. Compared to key hold latencies and key interval latencies, the availability of slur features was low even for 10-minute scans.

4.2 Authentication Results

Here, we report the verification (1:1) and identification (1: N) results.

4.2.1 Verification Results

Here, we report verification results with scaled Euclidean (SE), scaled Manhattan (SM), and relative distance (R) measure, with key hold (KH) and digraph (D) features.

Table 6. Detection Time versus Equal Error Rates for Individual Verifiers and Fusion

Detection Time	Equal Error Rate			
	Fused SE [KH] + SM [D] + R [KH]	SE [KH]	SM [D]	R [KH]
2 – Min	0.0408	0.0918	0.1139	0.1422
3 – Min	0.0310	0.074	0.0870	0.1171
4 – Min	0.0283	0.0656	0.0726	0.1007
5 – Min	0.0310	0.0605	0.0679	0.0942
6 – Min	0.0337	0.0547	0.0626	0.0849
7 – Min	0.0322	0.0545	0.0572	0.0787
8 – Min	0.0352	0.0556	0.0585	0.0732
9 – Min	0.0413	0.0554	0.0605	0.0722
10 – Min	0.0477	0.0524	0.0677	0.0717

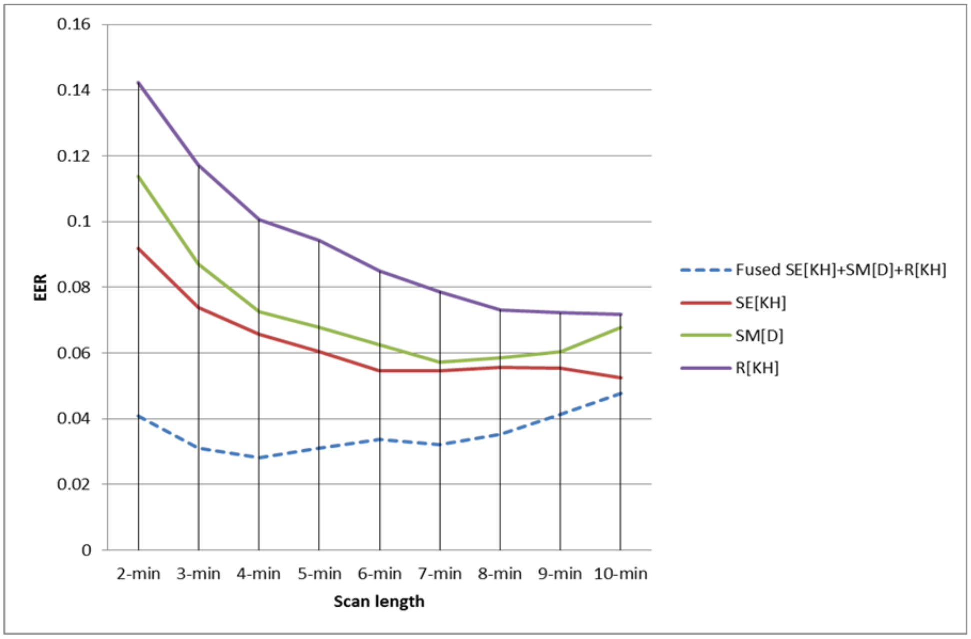


Figure 11. Detection Time versus Equal Error Rates for Individual Verifiers and Fusion

Table 7. Detection Time and Equal Error Rates of Fusion of Two Verifiers

Detection Time	Fused	Fused	Fused	Fused	Fused	Fused	Fused	Fused
	SE-KH	SE-KH	SE-D	SE-D	R-KH	R-D	R-KH	SM-KH
	+	+	+	+	+	+	+	+
	SM-KH	SM-D	SM-KH	SM-D	R-D	SM-KH	SM-D	SM-D
2-Min	0.0834	0.0463	0.1232	0.1623	0.0916	0.0561	0.0743	0.0493
3-Min	0.0688	0.0361	0.0984	0.1339	0.0650	0.0483	0.0496	0.0396
4-Min	0.0615	0.0321	0.0867	0.1115	0.0520	0.0407	0.0419	0.0355
5-Min	0.0595	0.0344	0.0874	0.1087	0.0454	0.0411	0.0403	0.0388
6-Min	0.0559	0.0349	0.0873	0.1022	0.0389	0.0380	0.0414	0.0393
7-Min	0.0549	0.0351	0.0889	0.1003	0.0341	0.0356	0.0401	0.0378
8-Min	0.0560	0.0385	0.0938	0.1024	0.0291	0.0349	0.0435	0.0427
9-Min	0.0543	0.0423	0.0923	0.1038	0.0296	0.0367	0.0480	0.0461
10-Min	0.0519	0.0488	0.0955	0.1043	0.0298	0.0319	0.0521	0.0544

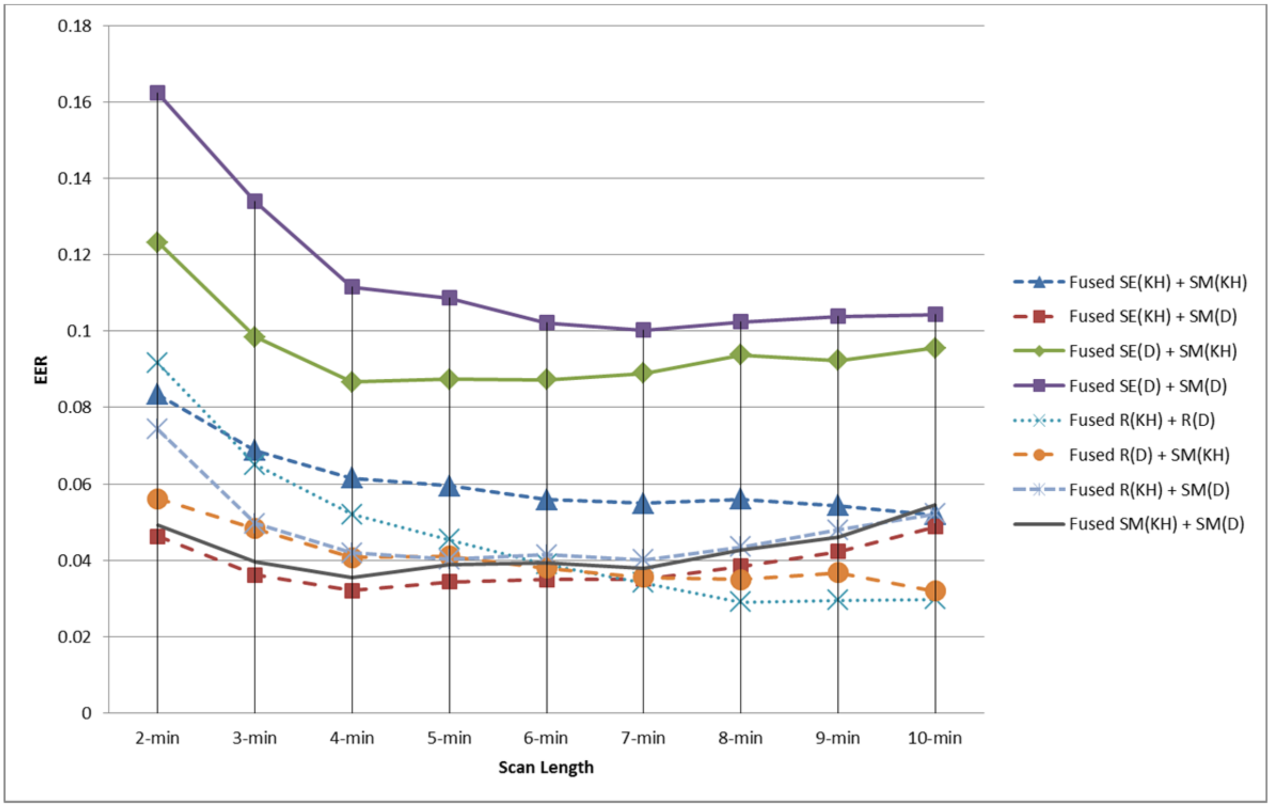


Figure 12. Detection Time versus Equal Error Rates of Fusion of Two Verifiers

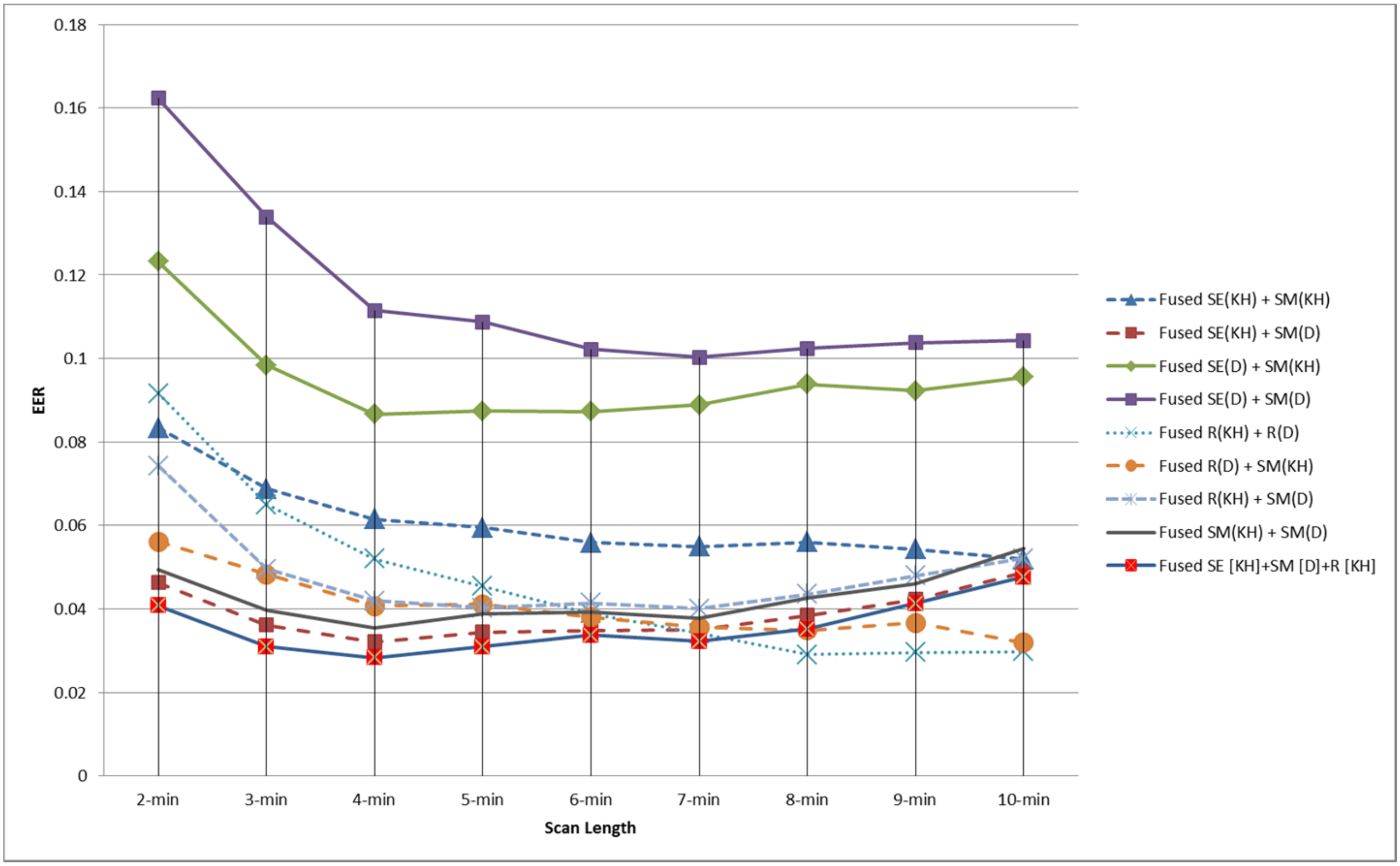


Figure 13. Comparisons of EERs with Fusion of Two and Three Verifiers

Discussion:

In Table 6 and Figure 11, we show the equal error rates of individual scaled Euclidean verifier with key hold features, scaled Manhattan verifier with digraph features and relative "R" distance verifier with key hold features and compare their performance with fusion verifier, which fuses all the three verifier-feature configurations. For fusion, we used weighted sum rule with equal weights assigned to the scores from each verifier-feature configuration. From the equal error rates in Table 6 and Figure 11, we observe that: (1) the verification errors considerably decrease as the detection time (or scan length) increases from 2 to 10 minutes; and (2) overall, the fusion verifier outperforms individual verifiers by considerable margin for all the tested scan lengths.

In Table 7 and Figure 12, we show the equal error rates of verification, obtained by fusing combinations of two and three verifiers, operating with key hold and digraph features. From Table 7 and Figure 12, we observe that: (1) any fusion combination, which involved both key hold and digraph latencies performed better than fusion combinations which used only one type of feature (e.g., see the plot of scaled Euclidean and scaled Manhattan with digraph features in Figure 12); and (2) a combination of same type of verifiers (e.g., scaled Euclidean and scaled

Manhattan verifier) did equally well compared to diverse verifier combinations (e.g., scaled Manhattan and R distance verifier).

In Figure 13, we compared the equal error rates of three-verifier fusion with various two-verifier fusion configurations. Overall, we observed that for lower scan lengths, between 2 and 5 minutes, three-verifier fusion outperforms two-verifier fusion configurations. However, for higher scan lengths, between 6 and 10 minutes, the two-verifier fusion configurations performed on par with the three-verifier fusion.

4.2.1.1 Scan-based Evaluation of Verification

Recall that each of the 486 users participated in two typing sessions. A session spans approximately between 45 minutes and 2 hours of typing activity. We used these two sets to create 40-hour authentication segments. During training, each user's template built from Set I data was subjected to impostor attacks from samples provided by the other users. From these attacks, we set a threshold for which the impostor pass rate was *β* =20%, and used it as the classification threshold during the genuine testing process. For genuine testing, we used each user's data from Set II to attack the template built using data from Set I.

Each user's typing session was divided into scans that each have a size *α*, with a rejection registered if, *m*=5 consecutive scans are rejected. The values of *m* and *β* were set in accordance with the DARPA performance specifications for the Active Authentication program. We experimented with values of *α* between 1 and 10. However, we only report results corresponding to α values between 5 and 10, since they produced the best performance.

Table 8. False Rejects in Scan Based Evaluation Across Different Atomic Features

Scan Length (minutes)	**5**	**6**	**7**	**8**	**9**	**10**
Number of Scans	480	400	343	300	267	240
Number of Users	115	118	120	125	127	131
Key Hold (K)	15	9	13	11	8	5
Key Interval (I)	16	17	7	5	10	8
Slur KI (S)	16	15	12	12	5	9
Digraph (D)	17	13	8	7	5	9
Trigraph (T)	21	14	11	5	15	10
Fourgraph (F)	21	18	13	5	15	9
Pause After Word (PAW)	22	17	8	7	7	5
Pause Before Word (PBW)	22	20	8	6	7	6
Fusion (F)	17	10	13	6	11	9

Table **8** shows the number of false rejects for different scan lengths in minutes. A false reject is registered when 5 consecutive genuine rejections occur. For example, with a scan length of 5 minutes (Column #2), there was a total of 480 scans from a set of 115 users. The features —K, I, S, D, T, F, PAW, PBW, F, CPS, WPS and KPS— respectively registered 15, 16, 16, 17, 21, 21, 22, 22, 17, 26, 10,21, 17, 18, 26 false rejects. Results for the last three features are based on two data collection experiments, labeled Phase VI and Phase VII. Observe that fusion did not depict any clear performance advantage over many of the individual features.

Table 9 captures the same results, except that the number of false rejects have been scaled by the number of scans. For example, the value 0.03125 in the fourth row of the table is obtained by dividing the number of false rejects in Table 8 (i.e., 15) by the number of scans (i.e., 480). This value gives a measure of the rate at which false rejections occur, while the corresponding value in Table 9 represents the false rejections in absolute terms.

Table 9. False Rejects per Scan in Scan-based Evaluation

Scan Length (minutes)	**5**	**6**	**7**	**8**	**9**	**10**
Number of Scans	480	400	343	300	267	240
Number of Users	115	118	120	125	127	131
Key Hold (K)	0.03125	0.0225	0.037901	0.036667	0.029963	0.020833
Key Interval (I)	0.033333	0.0425	0.020408	0.016667	0.037453	0.033333
Slur KI (S)	0.033333	0.0375	0.034985	0.04	0.018727	0.0375
Digraph (D)	0.035417	0.0325	0.023324	0.023333	0.018727	0.0375
Trigraph (T)	0.04375	0.035	0.03207	0.016667	0.05618	0.041667
Fourgraph (F)	0.04375	0.045	0.037901	0.016667	0.05618	0.0375
Pause After Word (PAW)	0.045833	0.0425	0.023324	0.023333	0.026217	0.020833
Pause Before Word (PBW)	0.045833	0.05	0.023324	0.02	0.026217	0.025

Discussion: From Table 8 and Table 9, we make the following observations:

- For the majority of features, the short scan lengths (e.g., α =5, 6) had higher numbers of false rejections than the longer scan lengths. This could be because the shorter scans encompass a very small number of features per authentication attempt, resulting into a reduced ability to uniquely identify users. In general, a longer scan should contain a higher number of characters (and hence features), which in turn provides a greater amount of evidence about the user during each authentication attempt.

- The longest scan lengths did not necessarily result into the best performance (e.g., a scan length of 10 does not necessarily result in better performance than a scan length of 8 across all features/verifiers). Although long scan lengths guarantee a high number of characters (and hence features) for user identification, they also result into a small number of authentication attempts for any given size of typing sample. With a reduced number of authentication attempts, it is difficult to get a stable representation of the user's behavior.

- In the majority of cases, fusion of the best verifier-feature combinations (see the row labeled Fusion (F) in both tables) resulted into improved performance relative to the individual verifiers.

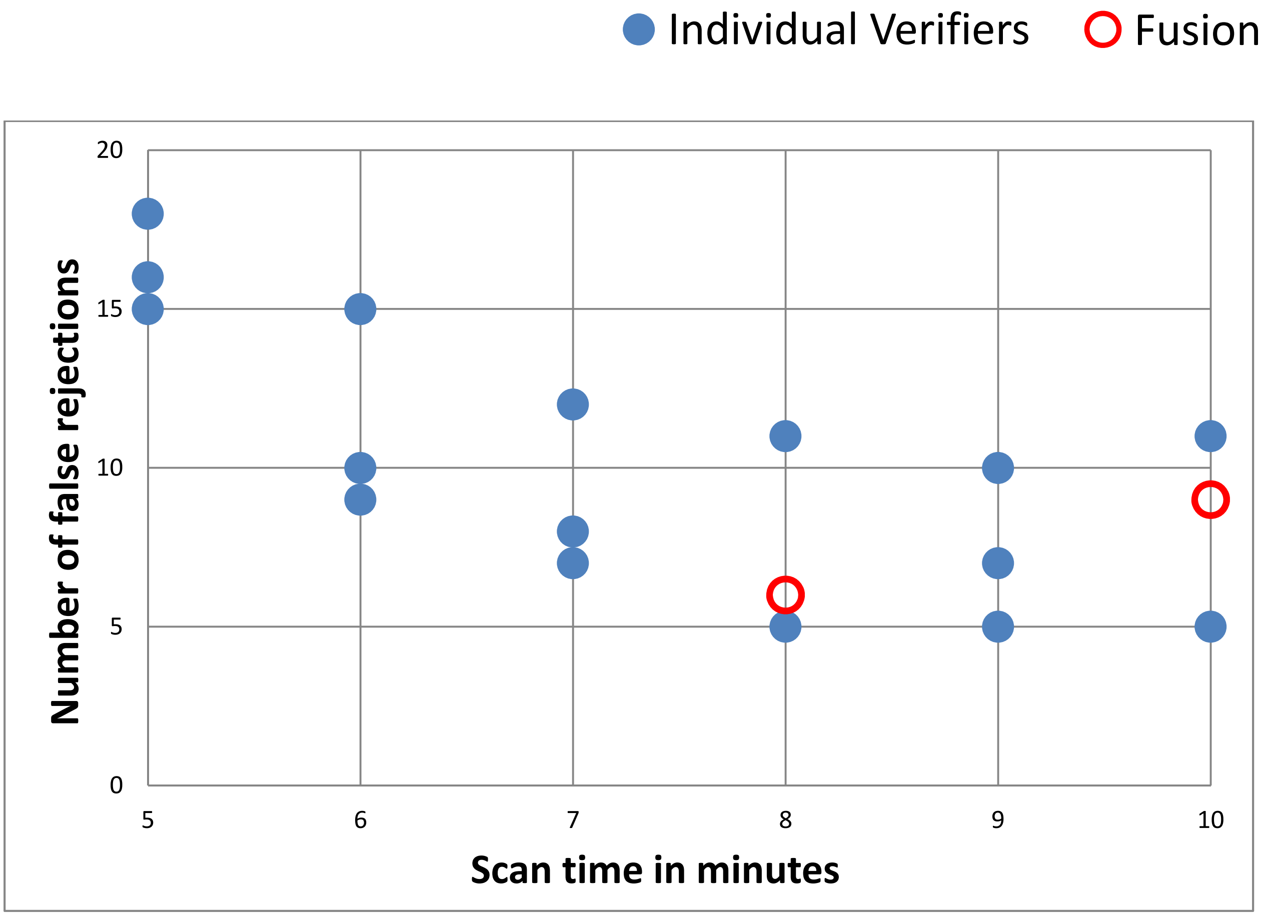


Figure 14. Scan-based Evaluation Results for 5 - 10 minutes scans.

Figure 14 gives a graphical view of the scan-based evaluation results. The full population of users is divided into 3 authentication segments based on the number of users required to realize 40 hours of scans. Each point plotted on the graph represents the minimum number of false rejects in a segment across all verifiers and features. The figure helps give a clearer view of the behavior highlighted in Tables 2 and 3. Observe that for the longer scans the false rejects were generally fewer in number. Also, the scan length of 10 minutes did not result into the lowest number of false rejects — The 9 minute scan length performed at least as well as any of the other scan lengths.

Further details on scan-based evaluation and results can be found in our paper published in IEEE IT Pro magazine.

4.2.1.2 Subsets Bootstrap Evaluation of Verification Results with Individual Verifiers

To estimate the 95% confidence intervals of the FRR and FAR, we used the subsets bootstrap procedure. This method, described in detail in [14], accounts for the dependencies between authentication attempts, and hence gives more accurate error estimates than the traditional parametric and bootstrap methods used for the same purpose.

Figure 15, Figure 16, and Figure 17 show the variation of the False Reject Rate (FRR) and False Acceptance Rate (FAR) with the classification threshold for various scan lengths and verifier-feature combinations. Reported results are the best across all verifier-feature combinations. The dashed lines on the either side of the full line represent the 95% confidence interval calculated using the subsets bootstrap procedure. From these results, we see that the best EERs registered with individual were between 0.2 and 0.4. These EERs are much higher than those reported in some past works (e.g., [5]). We hypothesize that our much larger population size could be a major reason for this difference in performance.

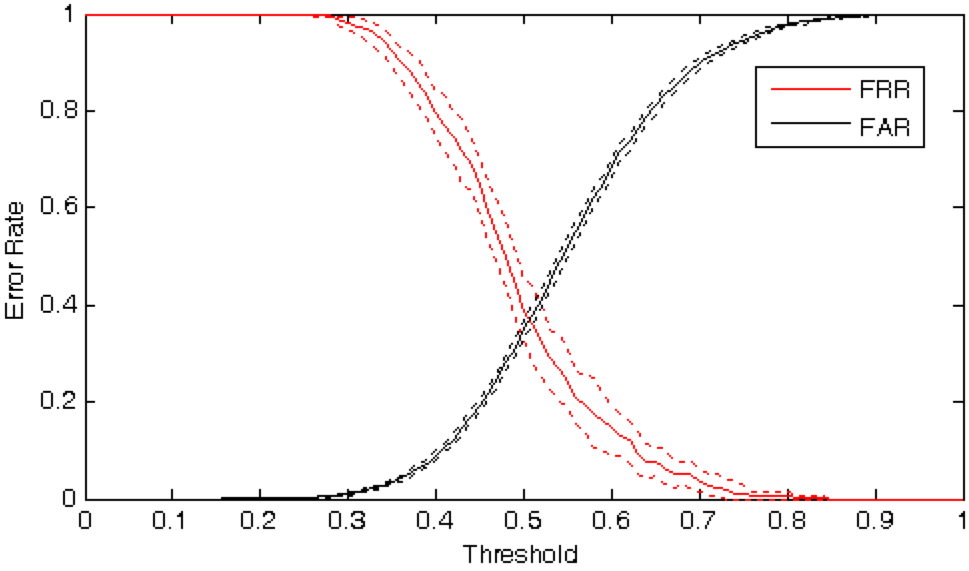


Figure 15. Variation of the False Reject Rate (FRR) and False Acceptance Rate (FAR) When Scan Length is 6 minutes; the Feature and Verifier are Respectively the Key Hold Time and Relative Verifier. The Dashed Llines Represent the 95% Confidence Interval Calculated Using the Subsets Bootstrap Procedure

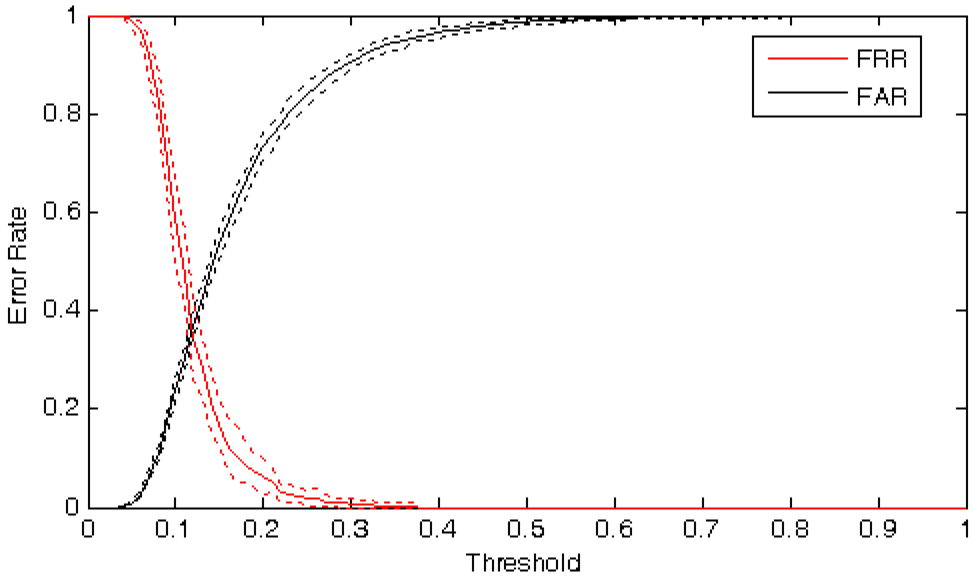


Figure 16. Variation of the False Reject Rate (FRR) and False Acceptance Rate (FAR) When Scan length is 7 Minutes; the Features and Verifiers are Respectively the Key Hold Time and Scaled Manhattan Verifier. The Dashed Lines Represent the 95% Confidence Interval Calculated Using the Subsets Bootstrap Procedure.

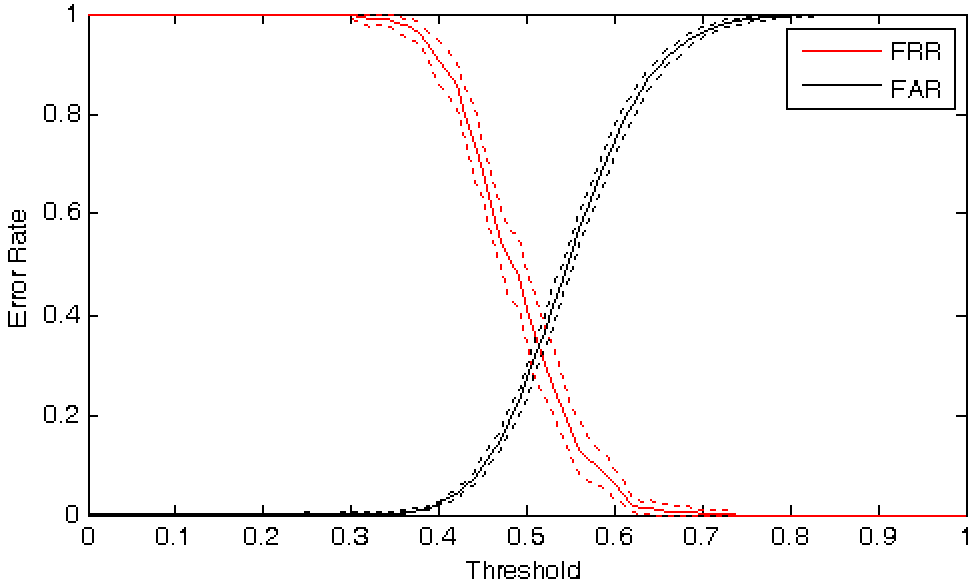


Figure 17. Variation of the False Reject Rate (FRR) and False Acceptance Rate (FAR) When Scan Length is 8 minutes; the Feature and Verifier are Respectively the Key Interval Time and Relative Verifier. The Dashed Lines Represent the 95% Confidence Interval Calculated Using the Subsets Bootstrap Procedure

4.2.2 Identification Results

In the following Figure 18, Figure 19, Figure 20, Figure 21, and Figure 22, we report the results of identification using key hold and digraph latencies with R distance and Scaled Manhattan distance identification systems. We also report the results of fusion of "R Distance with Digraph Features" and "Scaled Manhattan Distance with Key Hold Features." In the figures, the cross over points between false negative identification rate and false positive identification rate are highlighted with "red" circles. The crossover points show the threshold at which an identification system erroneously rejects "x" percentage of probe transactions from registered users while accepting "x" percentage of probe transactions from non-registered users.

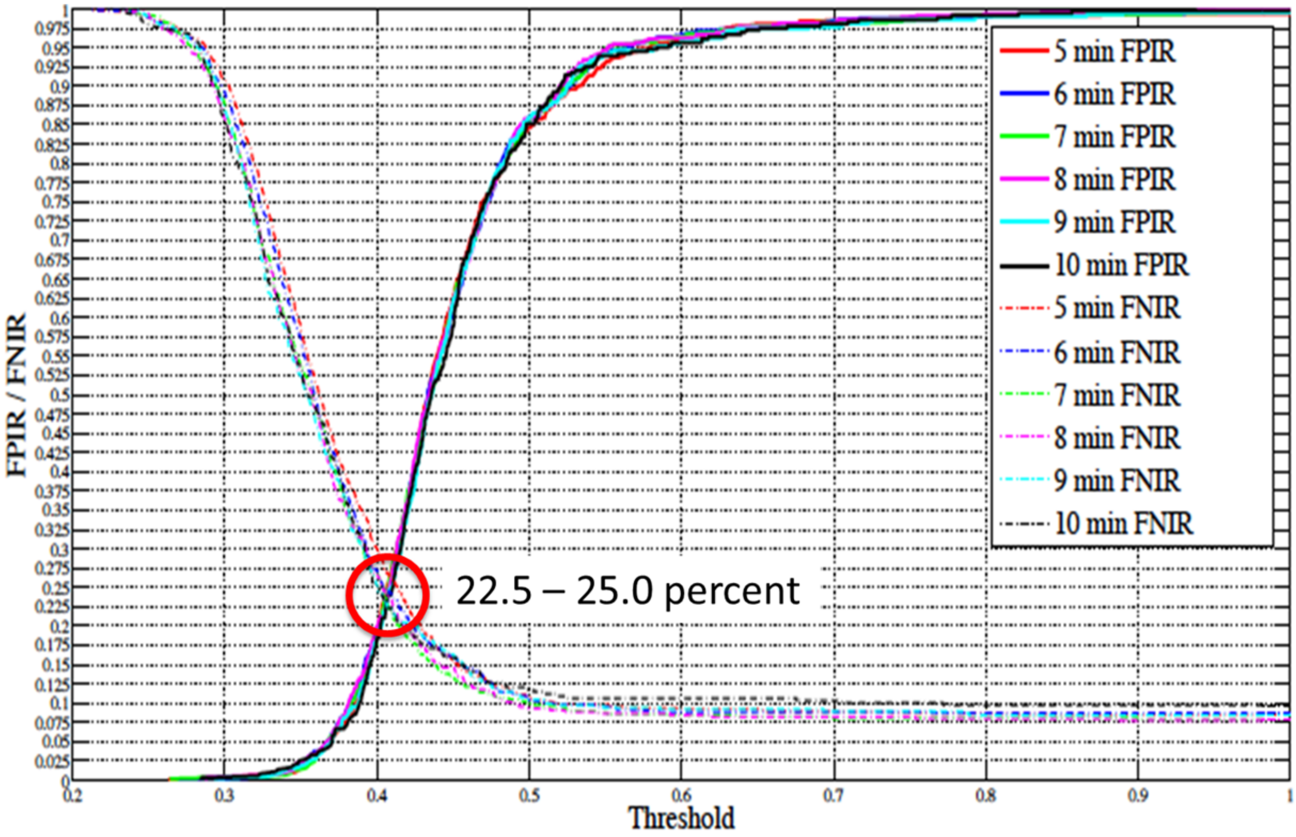


Figure 18. False Positive Identification Rate (solid lines) and Rank 5 False Negative Identification Rate (dotted lines) of Fusion based Identification System for Scan Lengths 5 minutes through 10 minutes. The Fusion was Between "R Distance with Digraph Features" and "Scaled Manhattan Distance with Key Hold Features". The Cross-over Points between False Positive Identification Rates and False Negative Identification Rate Occurred between 22.5 Percent and 25.0 Percent

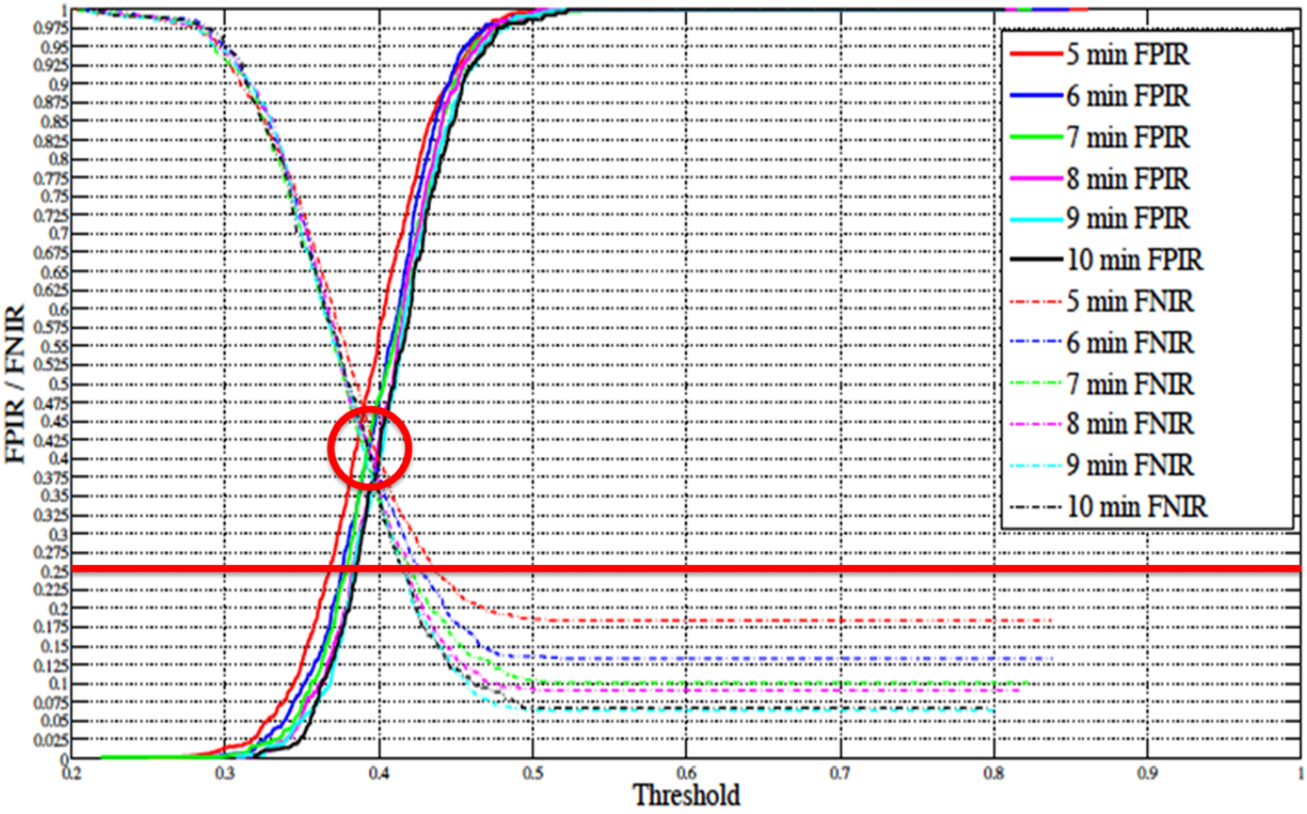


Figure 19. False Positive Identification Rate (solid lines) and Rank 5 False Negative Identification Rate (dotted lines) of "R Distance" Identification System with "Digraph Features" for Scan Lengths 5 Minutes through 10 Minutes. The Cross-over Points between False Positive Identification Rates and False Negative Identification Rate Occurred between 40.0 Percent and 45.0 Percent. The "Red" Horizontal Line Shows the Crossover Points Achieved with Fusion

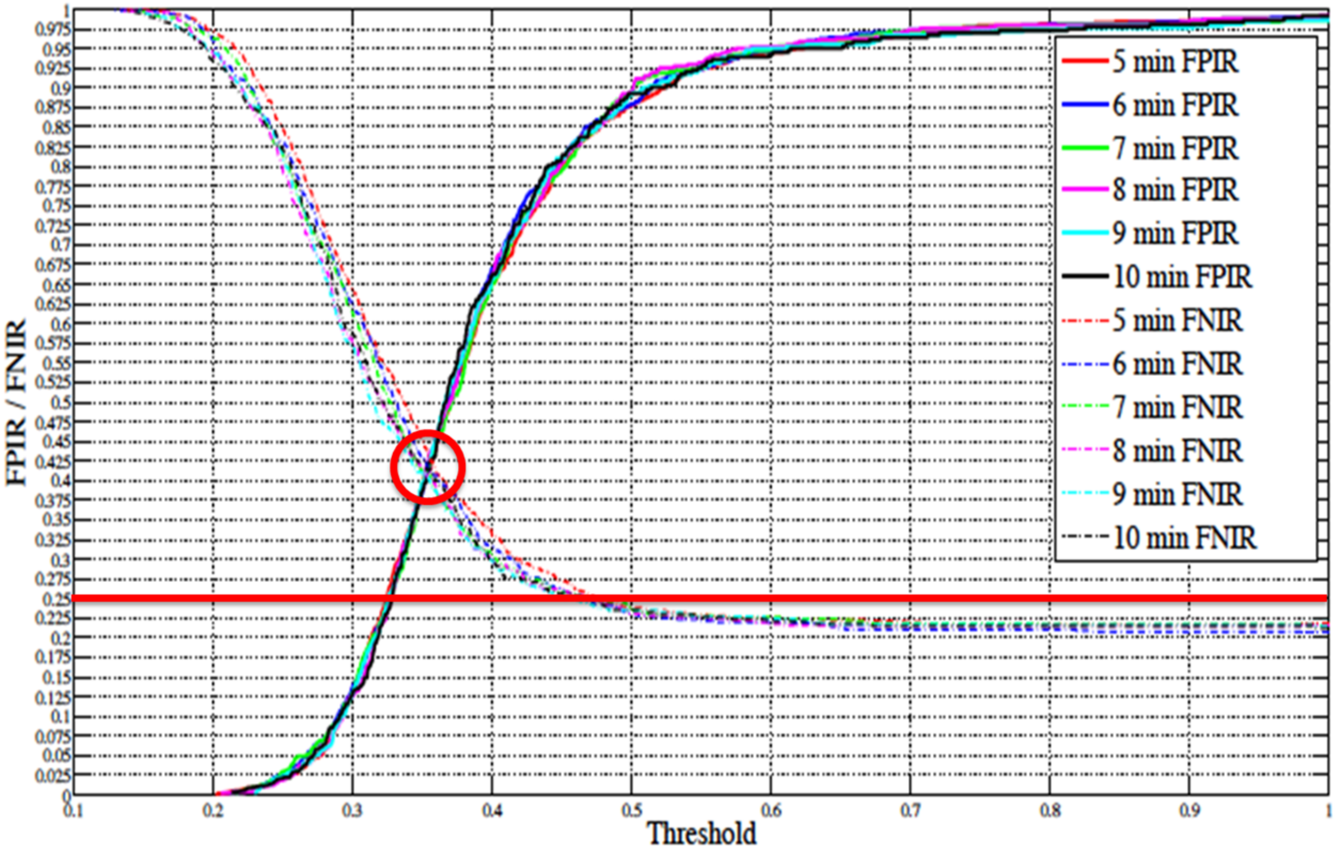


Figure 20. False Positive Identification Rate (solid lines) and Rank 5 False Negative Identification Rate (dotted lines) of "Scaled Manhattan Distance" Identification System with "Key Hold Features" for Scan Lengths 5 Minutes through 10 Minutes. The Crossover Points between False Positive Identification Rates and False Negative Identification Rate Occurred between 40.0 Percent and 45.0 Percent. The "Red" Horizontal Line Shows the Crossover Points Achieved with Fusion

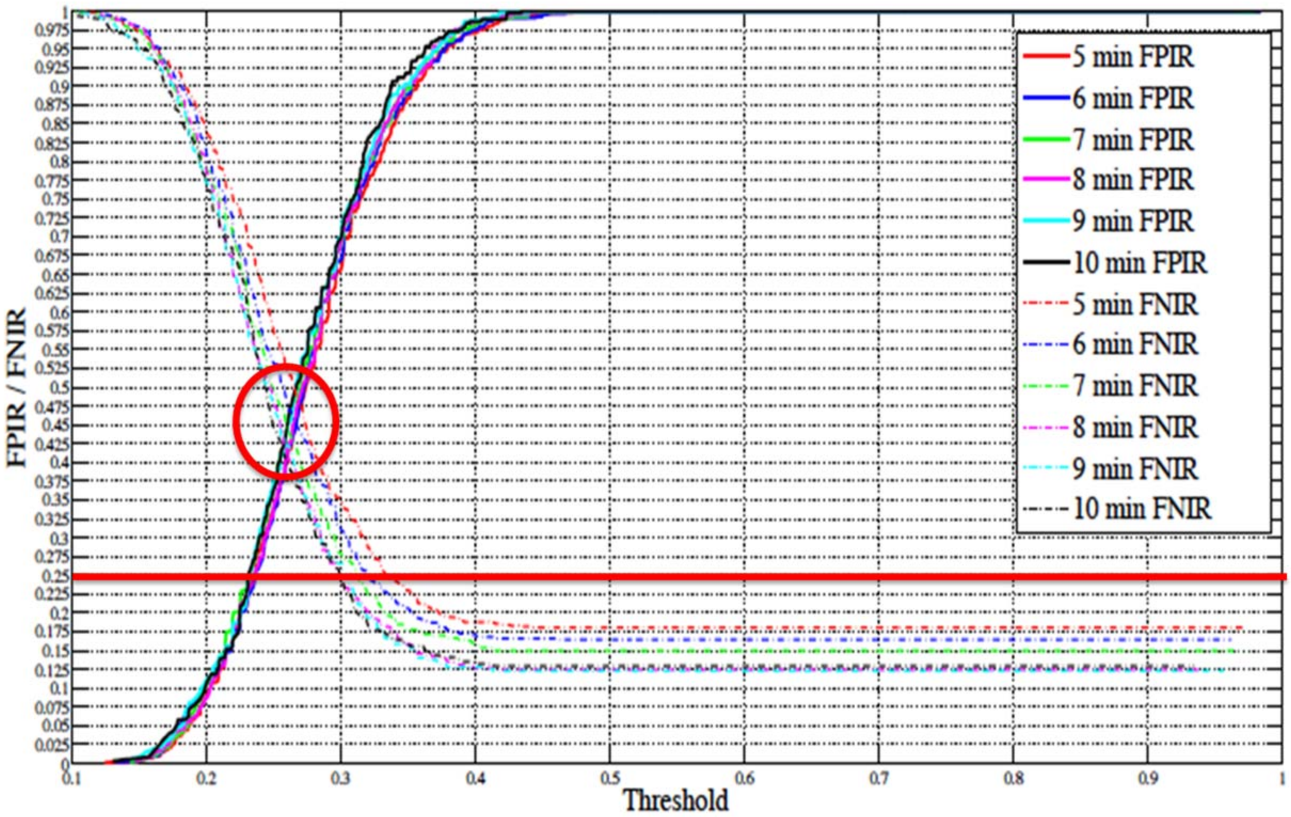


Figure 21. False Positive Identification Rate (solid lines) and Rank 5 False Negative Identification Rate (dotted lines) of “R Distance” Identification System with “Key Hold Features” for Scan Lengths 5 minutes through 10 minutes. The Crossover Points between False Positive Identification Rates and False Negative Identification Rate Occurred between 40.0 Percent and 50.0 Percent. The “Red” Horizontal Line Shows the Crossover Points Achieved with Fusion

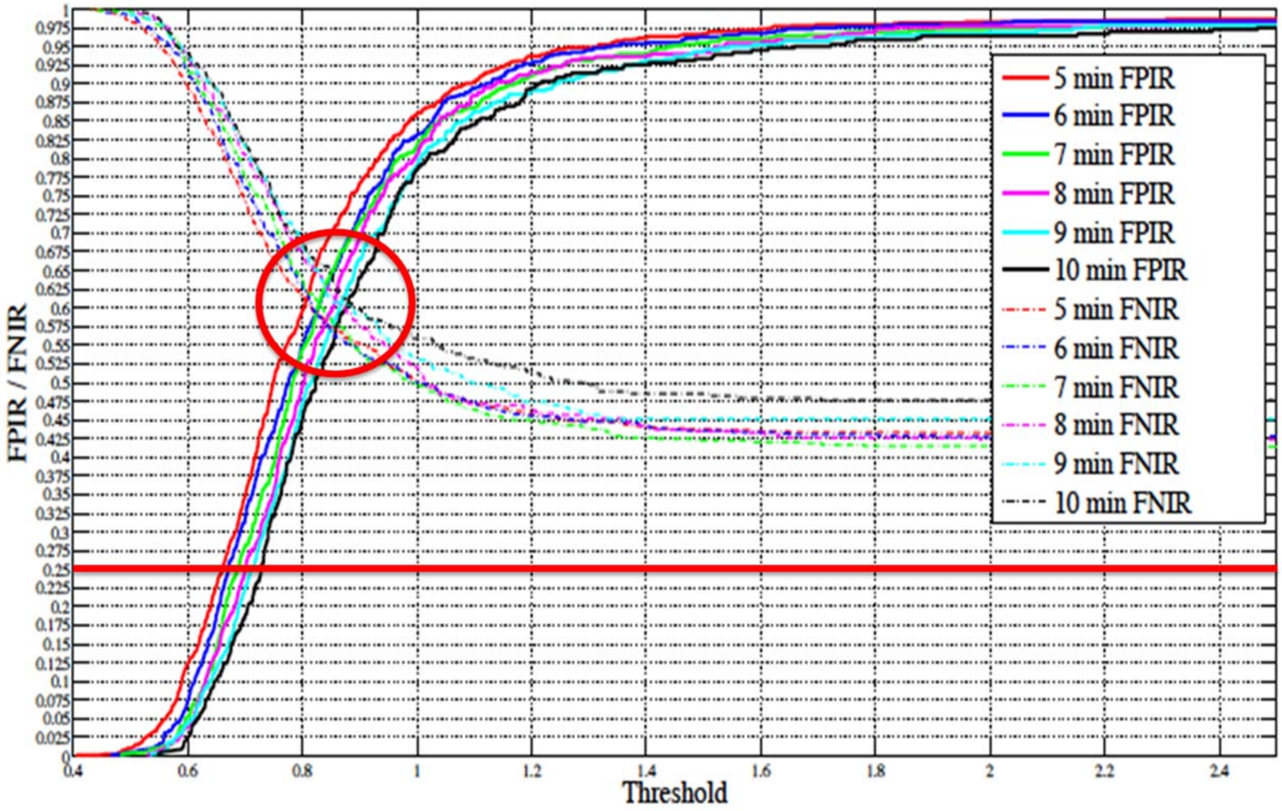


Figure 22. False Positive Identification Rate (solid lines) and Rank 5 False Negative Identification Rate (dotted lines) of "Scaled Manhattan Distance" Identification System with "Digraph Features", for Scan Lengths 5 minutes through 10 minutes. The Crossover Points between False Positive Identification Rates and False Negative Identification Rate Occurred between 60.0 Percent and 70.0 Percent. The "Red" Horizontal Line Shows the Crossover Points Achieved with Fusion

Discussion:

From Figure 18, Figure 19, Figure 20, Figure 21, and Figure 22, we observe that the fusion based identification system significantly outperforms individual identifications systems. While crossover points between 22.5 percent and 25.0 were achieved with fusion based identification system, we achieved between 40.0 percent and 67.5 percent cross over points individual metric based identification systems.

4.3 Snoop-forge-replay Attack Results

Here we highlight the error rates of snoop-forge-replay attack and contrast them with baseline (zero-effort) error rates. Detailed results can be found in our published journal paper.

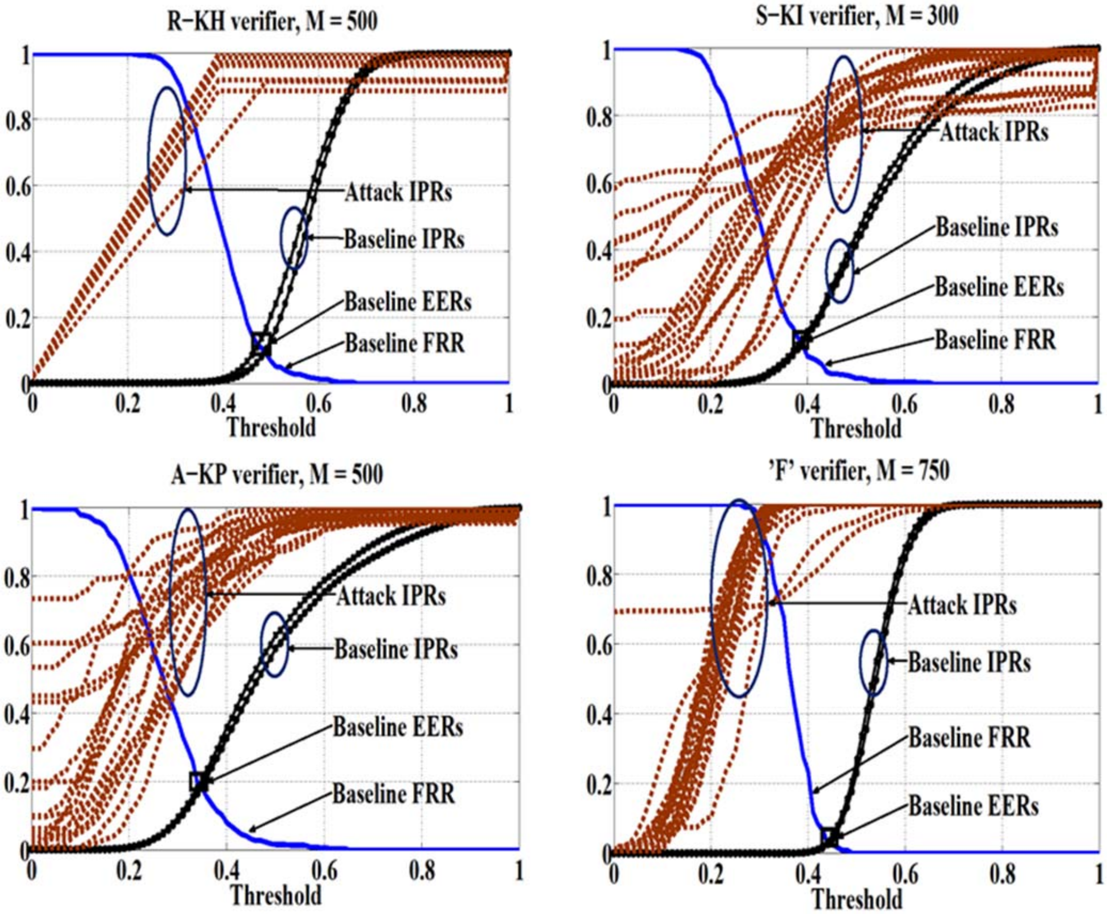


Figure 23. Comparison of Error Rates Achieved with Snoop-forge-replay Attacks Against "R" Distance Verifier, "S" Verifier, and "A" Verifier, Compared to the Baseline Error Rates with Zero-Effort Attacks

Figure 23 shows the detection error tradeoff (DET) curves obtained with Relative (R) distance verifier, Absolute (A) distance verifier, and S (Similarity) verifier with key hold (KH), key interval (KI), and key press (KP) latencies. The baseline (zero-effort) impostor pass rate curves are depicted in "black" color. The impostor pass rate curves of snoop-forge-replay attacks are depicted in "brown" color. From Figure 23, it is clear that the impostors pass rates with snoop-forge-replay attacks are significantly higher than baseline impostor pass rates.

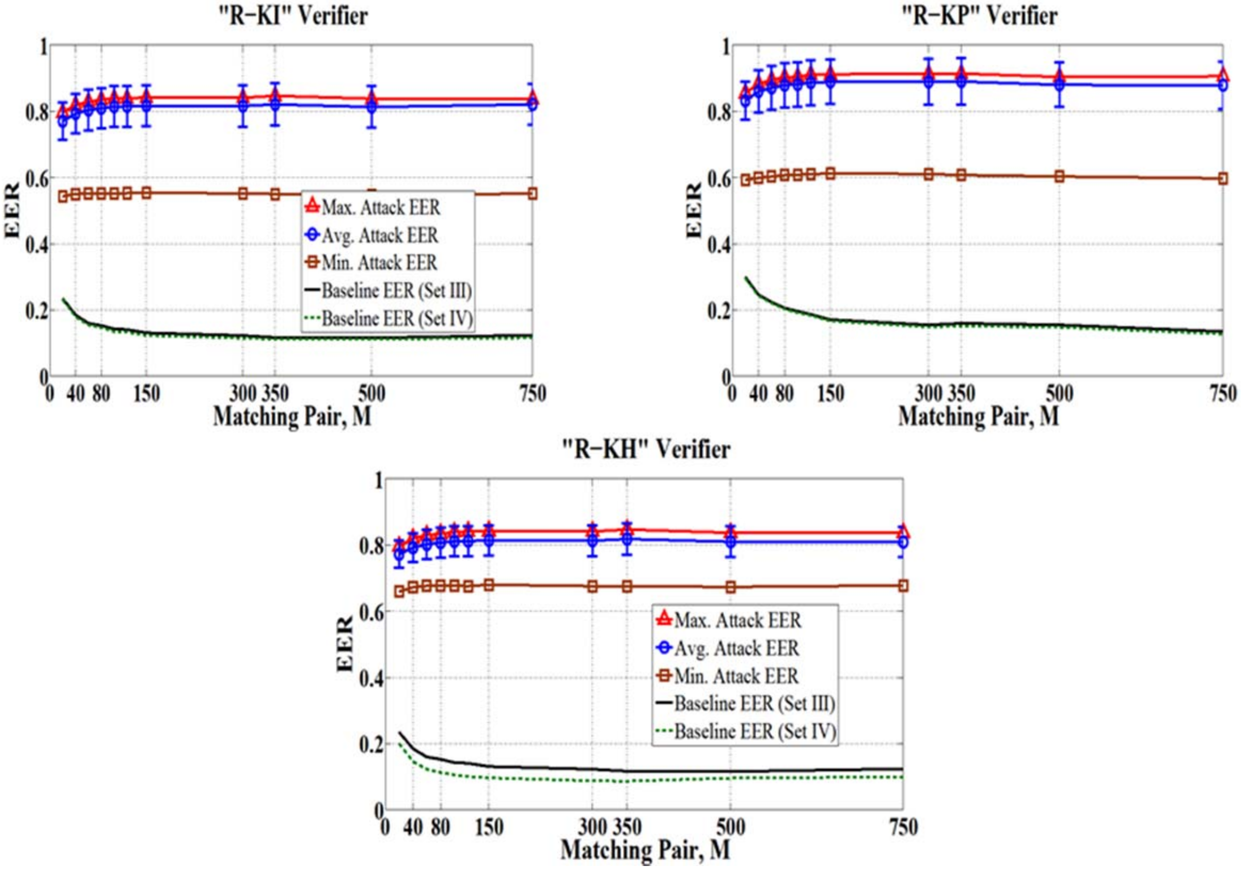


Figure 24. Equal Error Rates Achieved with Snoop-Forge-Replay Attacks Against R Distance Verifier with Key Interval, Key Press, and Key Hold Latencies as Features. The Baseline Equal Error Rates with Zero-Effort Attacks are Also Given for Comparison. The X-axis Represents the Number Of Matching Pairs Used in each Verification Attempt and the Y-Axis Represents the Equal Error Rate.

Figure 24 shows the differences in baseline EERs obtained with zero-effort impostor attacks and EERS obtained with snoop forge replay attacks on R verifier with key hold, key interval, and key press latencies as features. The plots in clearly show that, for every matching pair configuration, the equal error rates of snoop-forge-replay attacks are considerably higher than the baseline equal error rates (the black and green lines at the bottom of each plot in Figure 24).

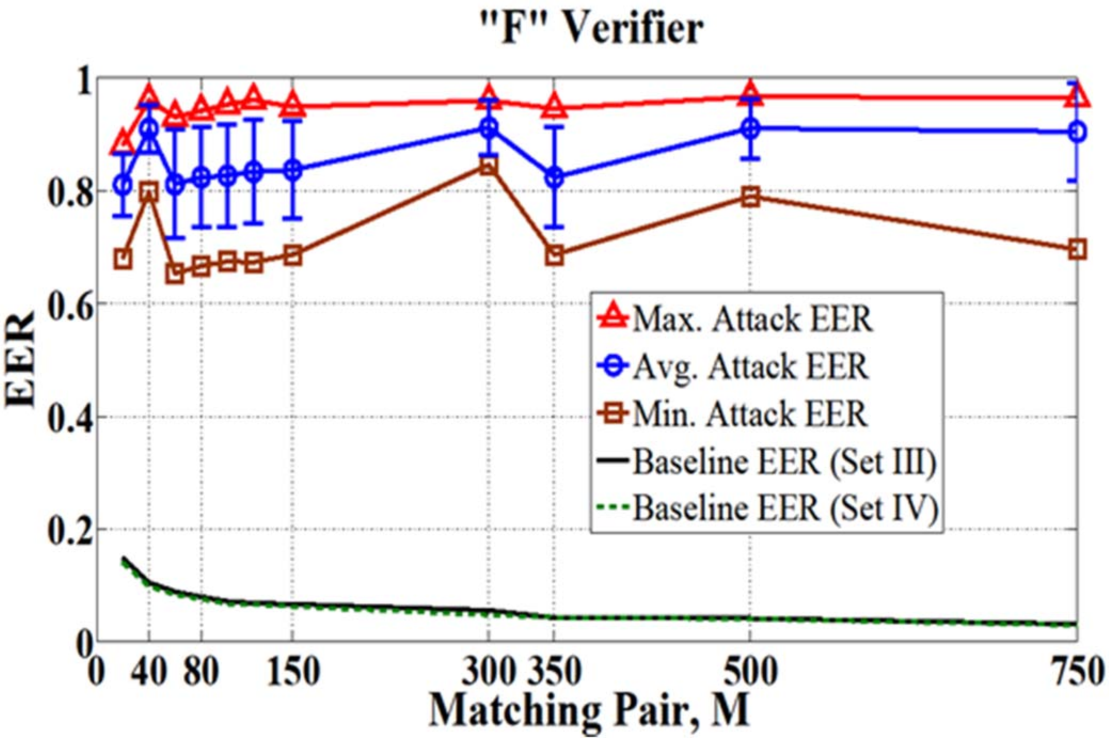


Figure 25. Equal Error Rates Achieved with Snoop-Forge-Replay Attacks Against Fusion (F) Distance Verifier. The Baseline Equal Error Rates with Zero-Effort Attacks are Also Given For Comparison. The X-Axis Represents the Number Of Matching Pairs Used in Each Verification Attempt and the Y-Axis Represents the Equal Error Rate.

Figure 25 shows the differences in baseline EERs obtained with zero-effort impostor attacks and EERs obtained with snoop-forge-replay attacks on fusion verifier. The plots in clearly show that, for every matching pair configuration, the equal error rates of snoop-forge-replay attacks are considerably higher than the baseline equal error rates (the black and green lines at the bottom of each plot in Figure 25).

Discussion:

With snoop-forge-replay attacks, we achieved as high as 125.5 to 2915.62 percentage increase in error rates compared to the error rates with baseline zero-effort impostor attacks. Our results reveal that there is a wide disparity in the error rates achieved with the zero-effort impostor attacks and the error rates achieved with snoop-forge-replay attacks. The high error rates with snoop-forge-replay attacks raise two fundamental questions: 1) is it secure to use keystrokes to continuously authenticate computer users? and 2) how can we redesign keystroke based continuous authentication systems that are resilient to forgery attacks?

We also observed that snooping more keystrokes from a victim user does not necessarily result in better attacks. In fact, our results with two verifiers (S and A) showed that snooping more keystrokes decreased the pass rates of the attacks. In our journal paper, we analyzed why snooping more keystrokes may have adversely affected the attack performance. Our results also showed that filtering outliers in the snooped keystrokes and considering digraphs that have occurred at least twice improved the pass rates of the attack. Gaussian perturbation made the attack weak against S and A verifiers and had least effect on R and F verifiers.

Our work demonstrates that the attacker, by exploiting virtualization, can reduce the time to forge the number of attacks by using more virtual machines. By increasing the number of virtual machines, the attacker can also generate a huge number of forgeries (e.g., in the order of millions) or scale the attack to victimize thousands of users.

4.4 Frog-boiling Attack Results

To evaluate the performance of the attack, we used the scaled Manhattan verifier (SM) and the Relative (R) distance verifier. We also studied fusion configuration of these two verifiers. Table 10 shows the mean and standard deviation of the EERs of the three verifiers *before* the attack was launched (i.e., at baseline). Both the R and fusion verifiers have EERs of over 40%. The SM verifier performs the best with an EER of 19.07.

Table 10. The Mean and Standard Deviation of The EERs ff the Three Verifiers at Baseline. All EERs are Expressed on a Scale Of 0-100 (i.e., as Percentages)

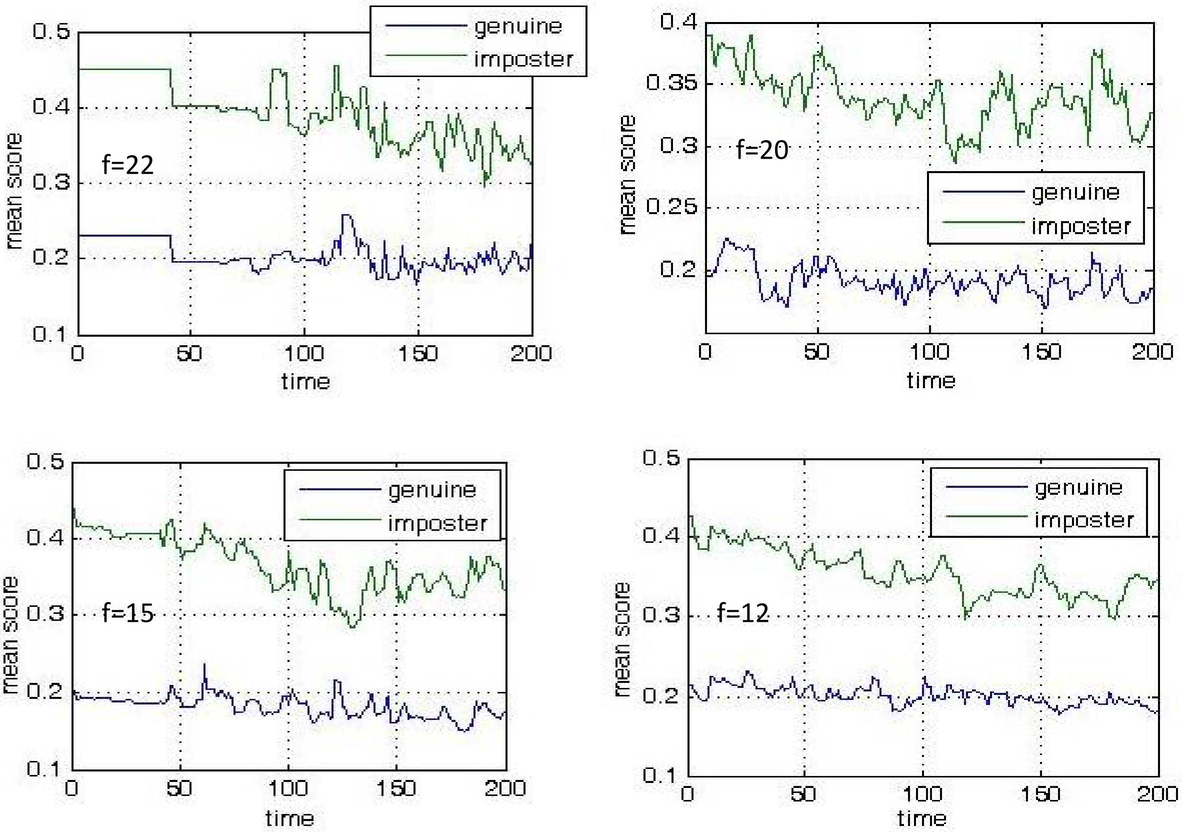
	SM	R	Fusion
Mean	19.07	46.97	42.61
SD	11.88	14.54	13.75

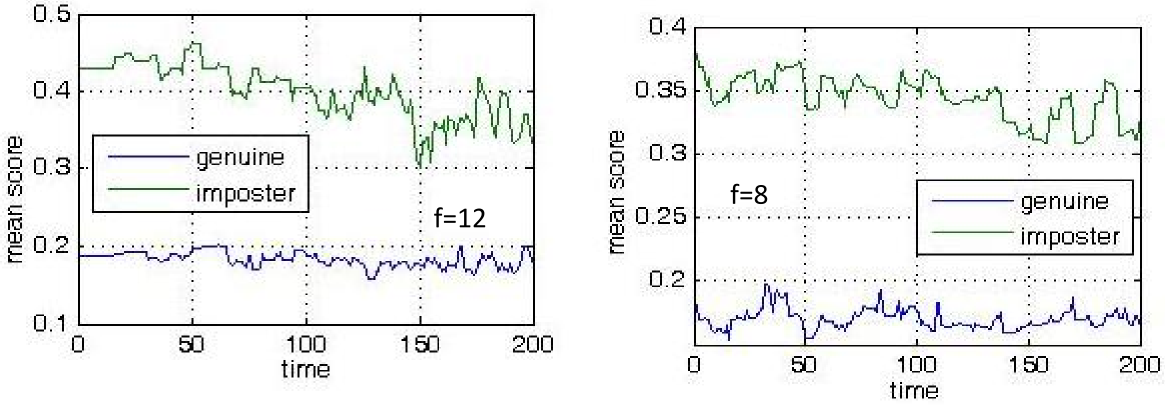
Table 11 shows the EER increments after the frog-boiling attack (due to its good performance, only SM verifier is tested against the frog-boiling attack in this case). When the genuine to imposter sample arrival rate is in the ratio 2:1, and arrivals follow a Uniform distribution, the frog-boiling attack performs the best (i.e., resulting in an ERR of 23.02%). When arrivals were Poisson, the mean EER under the attack was not much different. Compared to the results reported with fixed text however (see [12]), the attack has small effect on the performance of free text authentication. (i.e., it only increased the mean ERR from 19.07 to 22.51 at most, while the attack on the fixed-text system [12] saw EER increments that exceeded 50% of the baseline EER).

Table 11. The Mean EER after the Frog-Boiling Attack for the SM Verifier. "G" Represents the Number of Genuine Attempts; "I" Represents the Number of Imposter Attempts. For Each G:I Ratio, this Table Reports Results when: 1) Both Genuine and Imposter Arrivals Follow Poisson Distribution, and 2) Both Genuine and Imposter Arrivals Follow Uniform Distribution

	Poisson	Uniform
G:I = 1:2	22.51	20.64
G:I = 1:1	21.93	20.95
G:I = 2:1	22.07	23.02

Figure 26 shows the change of the genuine and impostor scores over time during the course of the frog-boiling attack. For the attack to have significant impact, the genuine scores should increase while the impostor scores increase over time. Compared to the scores evolution seen with fixed-text keystroke authentication (see [12]), these results confirm that the attack has a much reduced impact on free text authentication, since the genuine and impostor score distributions remain distinct from each other throughout the attack. We conjecture that the large number of features in each continuous authentication attempt (relative to a few features in a short password string) helps reduce the impact of the forgery. Another observation from Figure 26 is that the number of users who succumb to all 200 time units of attack is highest when the genuine:imposter arrival rate is 1:2 (22 and 20), and lowest when genuine/imposter arrival rate is 2:1 (12 and 8). This indicates that an attacker who accesses the system more frequently than the legitimate user can still have some negative effect on system performance.





(a) (b)

Figure 26. Change Of Mean Genuine (Blue) and Mean Impostor (Green) Scores Over Time When Sample Arrival Time Is (a) Poisson and (b) Uniform Distributed. The First, Second, And Third Rows Respectively, Show The Result When the Number of Genuine To Impostor Sample Ratio is 1:1, 1:2, And 2:1. Scores Plotted are Only for Users Who Succumb to 200-Time Units Of Attacks. The Number Of Such Users Is Represented by *f*.

4.5 Prediction of Cognitive Loads and Demographic Information

By linking the traditional stylometric qualities with production (timing, pausing, and revision) behavior, a richer understanding of a typist's cognitive load and demographic indicators is available. Table 12 includes results of prediction of cognitive load levels from cognitive rhythm features extracted from the typing data. Baseline performance and the classifier which achieved the best performance are reported. Because the classes are approximately balanced, we report performance in terms of accuracy.

Table 12. Cognitive Load Prediction Performance of Top Performing Classifier

Top Performing Classifier	Classification Task	N	Accuracy	Baseline
AdaBoost	1 vs. 6	2	75.00	50.00
SVM Linear Kernel (C=0.1)	1&2 vs. 5&6	2	68.89	55.55
L2-regularized Logistic Regression	1,2,3 vs. 4,5,6	2	60.17	58.33
SVM Linear Kernel (C=0.1)	1,2 vs. 3,4 vs. 5,6	3	51.42	41.67
SVM Linear Kernel (C=0.1)	1 vs. 2 vs. 3 vs. 4 vs. 5 vs. 6	6	36.25	25.00

Discussion: The results in Table 12 indicate that we can distinguish high vs. low cognitive load with quite high accuracy. However, this performance suffers when we include levels 3 and 4 in the classification. The two way classification which includes these is only 1.84% higher than the baseline, compared to the 1vs.6 classification which is 25% higher. Performance remains high

(68.89%) when we include levels 2 and 5 in the 1&2 vs. 5&6 classification. Unsurprisingly as we move to finer grained distinctions performance suffers significantly. While the prediction remains higher than the baseline, the 6-way classification performance is only 36.25% accurate.

Table 13. Demographic Prediction Performance of Top Performing Classifier

Classifier	Task	N	F-Measure	Baseline (F)	Accuracy	Baseline (Accuracy)
AdaBoost	Gender	2	.596	.333	61.00	66.66
AdaBoost	Handedness	2	.288	.070	88.00	93.00
AdaBoost	Native Language	2	.692	.170	78.00	83.00

Table 13 includes results of Demographic Classification Experiments. Baseline performance and the classifier which achieved the best performance are also included. Because the classes are skewed, we report performance in terms of accuracy and F-measure, with corresponding baselines for each. F-measure is the harmonic mean of the precision and recall of the minority class. This measure is typically used in detection tasks and effectively assesses the ability of a classifier to identify tokens of the rare class.

Discussion: On every demographic prediction task we were able to exceed the F-measure baseline. Note that this is the measure that we are optimizing. This is most notable for "nativeness" classification. Future work in this will involve recognizing specific native languages rather than English vs. Non-English. Other experiments can yield better accuracy, but correctly identifying the majority class is less useful than effectively detecting outliers. We are continuing to investigate ways to improve the recognition of handedness. It is possible that the collected material contains too few left-handed users to effectively generalize their behavior.

5 CONCLUSIONS

We reported the results of five key research tasks: (1) designing and extracting more than 45 types of cognitive rhythms (atomic, pausality, linguistic, and revision features) that have enabled us to extract over 9000 individual features from text production data; (2) determining the availability and discriminability of cognitive rhythm features; (3) determining continuous verification and identification error rates of cognitive rhythm features over a population 486 users; (4) investigating the impact of snoop-forge-replay and frog boiling attacks against continuous verification; and (5) prediction of users' cognitive load levels and demographic information from typing data. Our key results are enumerated below.

(1) We performed availability and discriminability analysis of features. Atomic level features (especially, key hold features) had the highest hit ratio for the investigated scan lengths of 1-, 5-, and 10-minutes. Our main findings from availability and discriminability analysis include: (a) key hold and interval features outperformed trigraphs and "slur" features in terms of hit ratios and symmetric uncertainty; and (b) from a feature space comprising of 9000 features, only 32 key hold features, 160 key interval features, 6 trigraph features, 23

"slur" key interval features, and 10 "slur" trigraph features were recovered in 10-minute scans, indicating the highly "sparse" availability of atomic features for scan based authentication.

(2) Among the tested features and matching algorithms, we achieved the lowest verification (1-to-1 match) and identification (1-to-N match) errors with score-level fusion. We achieved the lowest verification equal error rates, between 2.83 and 4.77 percent, by fusing scaled Euclidean-Key hold, scaled Manhattan-Digraph, and R-Key hold verifiers. We achieved the lowest identification accuracies, between 22.5 and 25.0 percent (in terms of FPIR and FNIR crossover), when R-Digraph identifier was fused with scaled Manhattan-Key hold identifier.

(3) Our experiments with non-zero effort "snoop-forge-replay" attacks show that continuous verification systems relying on atomic features are susceptible to algorithmic forgeries created from snooped keystrokes. We achieved between 125.5 to 2915.62 percentage increases in equal error rates compared to the equal error rates with baseline "zero-effort" impostor attacks.

(4) In our experiments with "frog-boiling" attacks on keystroke based continuous verification systems, we were able to raise the equal error rate to 22.51 percent with frog-boiling attacks on scaled Manhattan verifier, compared to its baseline equal error rate of 19.07 percent. Our experiments reveal that, frog-boiling attacks are not as effective on continuous keystroke based verification systems compared to fixed keystroke (password based) verification systems.

(5) Depending on the granularity at which cognitive loads were predicted, we achieved classification accuracies between 75.0 and 36.25 percent for predicting cognitive load levels from cognitive rhythms. In addition, we achieved: an F-measure of 0.96 (compared to a baseline of 0.33) for predicting "gender", an F-measure of 0.288 (compared to 0.070) for predicting "handedness", and an F-measure of 0.692 (compared to baseline 0.170) for predicting "Is the native language of the typist English?" Our results show that cognitive rhythm features are good predictors of demographic indicators and cognitive load levels at courser levels of granularity.

BIBLIOGRAPHY

[1] D. R. Krathwohl, “A Revision of Bloom's Taxonomy: An Overview,” *Theory into Practice*, vol. 41, pp. 212-218, 2002.

[2] S. Strömqvist and E. Ahlsén. “The Process of Writing-A Process Report,” Gothenburg Papers in Theoretical Linguistics 83, Gothenburg, Sweden: University of Gothenburg, Department of Linguistics.

[3] M. A. Covington and J. D. McFall, “Cutting the Gordian Knot: The Moving-Average Type–Token Ratio (MATTR),” *Journal of Quantitative Linguistics*, vol. 17, pp. 94-100, 2010.

[4] M. A. Hall and G. Holmes, “Benchmarking Attribute Selection Techniques for Discrete Class Data Mining,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, pp. 1437-1447, 2003.

[5] D. Gunetti and C. Picardi, “Keystroke analysis of free text,” *ACM Transactions on Information and System Security,* vol. 8, pp. 312-347, 2005

[6] A. Jain and A. Ross, "Information Fusion in Biometrics," *Pattern Recognition Letters - Special issue: Audio- and video-based biometric person authentication,* vol. 24, pp. 2115-2125, 2003.

[7] J. Kittler, M.Hatef, R. P. Duin, and J. G. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, pp. 226 - 239, 1998

[8] A. K. Jain, A. Ross, and P. J. Flynn, *Handbook of Biometrics*: Springer, 2007.

[9] K. Nandakumar, Y. Chen, S. C. Dass, and A. K. Jain, "Likelihood Ratio-Based Biometric Score Fusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 30, pp. 342-347, 2008.

[10] K. A. Rahman, K. S. Balagani, V. V. Phoha, “Snoop-Forge-Replay Attacks on Continuous Verification With Keystrokes,” *IEEE Transactions on Information Forensics and Security*, Vol.8, pp.528-541, March 2013.

[11] M. Davies, “Word Frequency Data from the Corpus of Contemporary American English (COCA)” Feb. 2008 [Online]. Available: http://www.wordfrequency.info, Last accessed May 15, 2011

[12] W. Zibo, A. Serwadda, K. S. Balagani, and V. V. Phoha, "Transforming animals in a cyber-behavioral biometric menagerie with Frog-Boiling attacks," presented at Biometrics: Theory, Applications and Systems (BTAS), 2012 IEEE Fifth International Conference on, 2012.

[13] K. Killourhy and R. Maxion, "Why did my detector do that?!: predicting keystroke-dynamics error rates," in Proceedings of the 13th international conference on Recent advances in intrusion detection. Ottawa, Ontario, Canada: Springer-Verlag, 2010, pp. 256-276.

[14] R. M. Bolle, N. K. Ratha, and S. Pankanti, "Error analysis of pattern recognition systems: the subsets bootstrap," *Comput. Vis. Image Underst.*, vol. 93, pp. 1-33, 2004.

APPENDIX A –DATA COLLECTION QUESTIONS

Session 1

- List the recent movies you've seen or books you've read. When did you see or read them? What were they about? Please use complete sentences. (Cognitive Load: 1)
- Which sport(s) do you like to watch/play? If there are none that interest you, then what activities do you enjoy watching/playing? (Cognitive Load: 1)
- What made you decide to join Louisiana Tech University? (Cognitive Load: 1)
- Where is a place that you particularly enjoy visiting? Describe what makes you happy about being at this place. (Cognitive Load: 2)
- Describe a time when someone (at work, school, or home) really upset you. Explain how you handled this? (Cognitive Load: 2)
- What is your favorite place to go out for a meal? What do you like about this place? (Cognitive Load: 2)
- What would you do if you and a friend are on vacation alone and your friend's leg gets cut? Describe what the procedure you would use for first aid or for finding help. (Cognitive Load: 3)
- What would you do if you were home alone and a fire started? (Cognitive Load: 3)
- Explain what you think the difference is between "communicating with" someone and "talking to" someone. How are these two terms often confused? (Cognitive Load: 4)
- Compare and contrast two genres of music. (Cognitive Load: 4)
- What are the main differences with standardized tests such as the ACT, SAT, GRE, and TOEFL? Please describe some of the pros and cons of these tests. (Cognitive Load: 4)
- Mary lives alone in Manhattan where she owns the apartment she lives in as well as 5 floors of real-state. She works as an investment banker, and owns an estate in Aspen, Colorado. Robert lives in Dallas where he owns a home and a 12-story commercial building. He has three children and works in the family business. Who do you think is wealthier? Make your own assumptions to evaluate their wealth. (Cognitive Load: 5)
- What email provider do you think is the best (e.g., Yahoo, Hotmail, Gmail, Webmail, etc.)? Why and what improvements would you like to see in them? (Note: Do not disclose your email ID). (Cognitive Load: 5)
- What social networking web-sites do you use? Overall, would you say that social networks are a good or bad thing? Have they helped people and society or hurt it? Explain your opinion. (Cognitive Load: 5)
- Do you think it's a good idea to raise tuition for students in order to have money to make improvements to the University? Why or why not? (Cognitive Load: 5)
- Pretend a Hollywood executive offered to pay you to write and act in a movie. Create a movie plot with a character in it for yourself and remember that you will only be paid for creating an original plot to a movie. (Cognitive Load: 6)

- If you were to create a picture of any type of landscape you wanted what objects would you include in it? How would you go about creating the landscape, and what method would you use to make your landscape? (Cognitive Load: 6)
- How would you design your class if you were the teacher? What subject would you teach? How would you structure your tests, and grading? (Cognitive Load: 6)

Session 2

- What are some things that you like about Ruston? (Cognitive Load: 1)
- What are your favorite things about winter? (Cognitive Load: 1)
- What is the best thing you ever ate at a restaurant? Describe it. (Cognitive Load: 1)
- What would you say has been the best college class you have taken and what did you enjoy about that class? (Cognitive Load: 2)
- What is something that you dread talking to your family about? Why do you not like to talk to them about this? (Cognitive Load: 2)
- Can you describe the process of applying to college? If you didn't go to college, can you describe the process of applying for jobs? (Cognitive Load: 2)
- Suppose you were in NYC and had a very important presentation to give at 8AM the next morning at Louisiana Tech. You get to the airport in New York to discover that your flight has been delayed and will likely cause you to miss your layover in Atlanta. What steps would you take to insure that you are at Louisiana Tech in time for your presentation? (Cognitive Load: 3)
- What would you do if you woke up and realized your car would not start, but you had a very important meeting to attend at 9AM? (Cognitive Load: 3)
- Compare and contrast two sources you use for news and current events. This may include particular channels, newspapers, websites, or TV shows. (Cognitive Load: 4)
- Give step-by-step driving directions to your favorite place in or around Ruston, from Louisiana Tech University campus. Examples of favorite places could include restaurants, movie theaters, and other public places. (Note: Do not disclose private information such as home address.). (Cognitive Load: 4)
- Explain what the saying "Not all that glitters is gold" means. (Cognitive Load: 4)
- You own a company and need to promote someone to be the manager. Jim has a college degree, has been with your company for three years and has proven to be a strong team-player. He works well with others, and knows the company very well. Frank just started with your company, but has 20 years of experience with another company. He does not have a degree and struggles to get along with the other employees. (Cognitive Load: 5)
- Do you think that capital punishment should be legal? Why or why not? (Cognitive Load: 5)

- Many universities require that freshmen live in an on-campus apartment or dorm their first year of college. In your opinion, is this a good or bad thing? Explain your point of view on this topic. (Cognitive Load: 5)
- Do you think people should be required to have car insurance? Defend your decision. (Cognitive Load: 5)
- Discuss step-by-step instructions for a task (e.g., mechanical tasks such as cooking a specific dish, building or repairing something, etc.) that you are very familiar with. Write them so a person who has never done this before can follow your instructions. (Cognitive Load: 6)
- Decide on a party or event that you want to have and write details as to how you would plan this event. Write only about the planning you would do before the day of the event (budget, guest list, music, etc). (Cognitive Load: 6)
- Suppose you are the manager for a place on campus that sells school supplies, coffee, and food. The store is open 24 hours a day, 7 days a week. You have been put in charge of writing the new code of conduct for the hourly employees as well as developing a reward system to insure that they are coming to work on time and following procedure while they are there. Write out in detail what rules of conduct. (Cognitive Load: 6)

LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

API – Application programming interface

COCA – Corpus of Contemporart American English

CUNY – The City University of New York

DARPA – Defense Advanced Research Projects Agency

DET – Detection error tradeoff

EER – Equal error rate

FAR – False acceptance rate

FNIR – False negative identification rate

FPIR – False positive identification rate

FRR – False rejection rate

GUI – Graphical user interface

HR – Hit ratio

IDF – Inverse document frequency

LTU – Louisiana Tech University

MATTR – Moving-average type-token ratio

NYIT – New York Institute of Technolgy

OS – Operating system

PI – Principal investigator

SU – Symmetric uncertainty

TF – Term frequency